

МАШИННОЕ ОБУЧЕНИЕ ДЛЯ ДЕТЕЙ

Практическое введение
в искусственный интеллект

Дейл Лейн



МАШИННОЕ ОБУЧЕНИЕ ДЛЯ ДЕТЕЙ

Практическое введение
в искусственный интеллект





MACHINE LEARNING FOR KIDS

**A Project-Based Introduction to
Artificial Intelligence**



by Dale Lane



**no starch
press**

Дейл Лейн



МАШИННОЕ ОБУЧЕНИЕ ДЛЯ ДЕТЕЙ

Практическое введение
в искусственный интеллект



Электронное издание

Перевод с английского
М.А. Федотенко



Москва
Лаборатория знаний
2023

УДК 004.5
ББК 32.973.3 + 32.813.5
Л48

Серия основана в 2018 г.

Лейн Д.

Л48 Машинное обучение для детей. Практическое введение в искусственный интеллект / Д. Лейн ; пер. с англ. — Электрон. изд. — М. : Лаборатория знаний, 2023. — 291 с. — (Школа юного программиста). — Систем. требования: Adobe Reader XI ; экран 10". — Загл. с титул. экрана. — Текст : электронный.

ISBN 978-5-93208-646-9

Книга знакомит школьников с машинным обучением через выполнение 13 практических проектов в доступной образовательной онлайн-среде с применением языка визуального программирования Scratch. Все проекты в книге сопровождаются подробными пошаговыми инструкциями, доступными для любого новичка.

УДК 004.5

ББК 32.973.3 + 32.813.5

Деривативное издание на основе печатного аналога: Машинное обучение для детей. Практическое введение в искусственный интеллект / Д. Лейн ; пер. с англ. — М. : Лаборатория знаний, 2023. — 288 с. : ил. — (Школа юного программиста). — ISBN 978-5-93208-321-5.

6+



В соответствии со ст. 1299 и 1301 ГК РФ при устранении ограничений, установленных техническими средствами защиты авторских прав, правообладатель вправе требовать от нарушителя возмещения убытков или выплаты компенсации

Copyright © 2021 by Dale Lane.

Оригинальное англоязычное название:
Machine Learning for Kids: A Project-Based
Introduction to Artificial Intelligence,
ISBN 9781718500563, опубликовано
No Starch Press Inc. 245 8th Street, San
Francisco, California United States 94103.

© Перевод, Лаборатория знаний, 2023.

По лицензии No Starch Press Inc.

Все права защищены.

ISBN 978-5-93208-646-9

Об авторе

Дейл Лейн — ведущий разработчик в компании IBM с богатым опытом работы в сфере искусственного интеллекта и машинного обучения. Работал над многими проектами в области искусственного интеллекта для клиентов IBM, в течение нескольких лет был разработчиком IBM Watson Studio.

О техническом рецензенте

Майя Пош — разработчик программного и аппаратного обеспечения, специализируется на языках программирования C++, Ada и языке проектирования VHDL. Автор художественных и научно-популярных произведений.



https://t.me/it_boooks/2

Благодарности	10
Предисловие	11
Введение	13
Scratch	14
Знакомство с интерфейсом Scratch	14
Программирование в среде Scratch	15
Сохранение результатов вашей работы	17
Проект «Машинное обучение для детей»	18
Что дальше?	19
Глава 1. Что такое искусственный интеллект?	22
Программирование	22
Машинное обучение	23
Искусственный интеллект	24
Нейронные сети и глубокое обучение	25
Что вы узнали	26
Глава 2. Знакомство с онлайн-инструментом «Машинное обучение для детей»	27
Вход в систему	28
Создание нового проекта	29
Этапы работы над проектом	31
Этап «Обучить»	32
Этап «Узнать и проверить»	33
Этап «Создать»	34
Создание аккаунта	35
Что вы узнали	38
Глава 3. Сортировка изображений животных	39
Выполнение проекта	40
Обучение модели	40
Подготовка проекта	45
Тестирование модели	51
Обзор и улучшение проекта	52
Что вы узнали	55
Глава 4. Игра с компьютером в «Камень, ножницы, бумага»	56
Выполнение проекта	57
Обучение модели	57
Подготовка проекта-игры	61
Тестирование игры	64
Обзор и улучшение проекта	64
Что вы узнали	67
Глава 5. Распознавание постеров фильмов	68
Выполнение проекта	70
Обучение модели	70
Подготовка проекта	75



Тестирование модели	85
Обзор и улучшение проекта	86
Что вы узнали	86
Глава 6. Сортировка писем	87
Выполнение проекта	88
Обучение модели	89
Подготовка проекта	94
Тестирование модели	101
Обзор и улучшение проекта	103
Что вы узнали	103
Глава 7. Распознавание эмоций в тексте	104
Выполнение проекта	105
Подготовка проекта-игры	106
Программирование игры без использования машинного обучения	110
Обучение модели	112
Программирование игры с использованием машинного обучения	116
Тестирование игры	119
Обзор и улучшение проекта	119
Использование голосовых сообщений вместо напечатанных текстов	119
Распознавание речи, которая не содержит ни оскорблений, ни комплиментов	120
Обучение на ошибках	123
Что вы узнали	125
Глава 8. Распознавание стиля письма в газетных статьях	126
Выполнение проекта	127
Обучение модели	129
Подготовка проекта	132
Обзор и улучшение проекта	141
Оценка качества модели машинного обучения: показатель достоверности	142
Оценка качества модели машинного обучения: матрица ошибок	145
Оценка качества модели машинного обучения: точность и полнота	150
Улучшение вашей модели машинного обучения	152
Что вы узнали	152
Глава 9. Поиск объекта на картинке	154
Выполнение проекта	155
Обучение модели	156
Подготовка проекта	164
Тестирование модели	166
Обзор и улучшение проекта	168
Применение сложных систем распознавания изображений в реальных проектах	170
Что вы узнали	173

Глава 10. Умные помощники	174
Выполнение проекта	176
Создание программы без использования машинного обучения	176
Обучение модели	178
Создание программы с использованием модели машинного обучения	183
Тестирование модели	184
Обзор и улучшение проекта	185
Использование показателя степени уверенности вашей модели машинного обучения	185
Говорить, вместо того чтобы набирать текст вручную	188
Сбор обучающих примеров	189
Что вы узнали	189
Глава 11. Чатботы	191
Выполнение проекта	193
Подготовка персонажа	194
Обучение модели	195
Подготовка проекта	200
Тестирование модели	202
Обзор и улучшение проекта	202
Реакция на пользовательские сообщения об ошибках	203
Распознавание пользовательского недовольства	205
Отвечать на вопросы только тогда, когда модель уверена в своем ответе	206
Этические вопросы использования машинного обучения	207
Что вы узнали	209
Глава 12. Создание игры «Убег от монстра»	210
Выполнение проекта	211
Описание возможных состояний игры	213
Обучение модели	214
Тестирование игры	223
Обзор и улучшение проекта	226
Что вы узнали	229
Глава 13. Создание игры «Крестики-нолики»	230
Выполнение проекта	232
Подготовка проекта-игры	235
Обучение модели	247
Тестирование игры	250
Обзор и улучшение проекта	251
Что вы узнали	252
Глава 14. Запутать компьютер	253
Выполнение проекта	255
Обучение модели	257
Подготовка проекта	260
Тестирование модели	263
Обзор и исправление проекта	264
Что вы узнали	267

Глава 15. Этические вопросы использования искусственного интеллекта	268
Выполнение проекта.....	268
Обучение модели	269
Подготовка проекта.....	272
Тестирование проекта.....	274
Добавление предвзятости.....	275
Тестирование проекта с добавленной предвзятостью	278
Обзор проекта.....	279
Случаи, в которых предвзятость полезна	280
Искусственный интеллект и этические вопросы	281
Что вы узнали.....	283
Послесловие	284
Будущее машинного обучения	284
Что дальше?	285
Алфавитный указатель.....	287





Хочу выразить благодарность за помощь в создании этой книги сотрудникам издательства No Starch Press: директору издательства Биллу Поллоку, редакторам Барбаре Йен, Рэйчел Монаган, Патрику ДиХусто и Атабаске Витчи, а также техническому рецензенту Майе Пош.

Также я очень благодарен исследовательской группе Lifelong Kindergarten из Медиа-лаборатории Массачусетского технологического института (MIT) за изобретение Scratch, за его постоянное обновление и расширение функционала, а также за безвозмездное предоставление этого продукта любому пользователю (включая исходный код). Все это сделало возможным появление таких образовательных проектов, как «Машинное обучение для детей», книгу о котором вы сейчас держите в руках.

Отдельное спасибо Начальной школе Джона Кебла в Херсли за предоставленную мне возможность протестировать описанные в книге проекты, выполняя их с учениками этой школы.





Когда мой отец был ребенком, он рос в Иллинойсе. Это было на рубеже веков (только не в этом столетии, заметьте, а в предыдущем), когда было изобретено радио, когда только придумали застежку-молнию, когда Чарли Чаплин был популярен, а Чарльз Линдберг еще не перелетел Атлантический океан, когда затонул «Титаник», когда был выпущен легендарный автомобиль Ford Model T. Когда же я был ребенком и рос на Великих равнинах в Техасе, человек только высадился на Луну, компьютеры были огромными и дорогими чудовищами, Интернет еще не был изобретен, пластик только начал широко использоваться, а разговор с кем-то на другом конце света был непомерно дорогим для абсолютного большинства людей. Я провел много летних дней, мечтая о полете к звездам и готовя в своей спальне (прости, мама!) порох для запуска моделей ракет. Я собрал свой первый компьютер с нуля, когда мне было 12 лет, и, когда я говорю «с нуля», это именно то, что я имею в виду — из отдельных транзисторов, диодов, резисторов. Я читал о роботе по имени Шейки, которого собрали ребята из Стэнфордского научно-исследовательского института, и сидел в кинотеатре как загипнотизированный, когда видел компьютер HAL 9000 в фильме «2001 год: Космическая одиссея». Уже тогда я знал, что хочу создавать компьютеры, которые помогут людям добраться до звезд.

И вот он я сейчас, сделавший для этого достаточно много.

Сегодня мир сильно отличается от того, что мы с отцом видели в детстве. В чем-то он лучше, в чем-то нет, но есть одна вещь, которая точно не изменилась — это наша способность мечтать и работать над тем, чтобы мечты стали реальностью. Сегодня можно получить доступ к любым знаниям с помощью устройства, которое умещается на ладони. Можно общаться в режиме реального времени с кем-то, кто находится на другом конце мира или даже в космосе. Автомобили теперь электрические, а некоторые из них даже ездят без водителя. Человечество не только побывало на Луне, но и отправило множество роботов на Марс, а несколько космических кораблей исследует межзвездное пространство за пределами нашей Солнечной системы. Благодаря компьютерам мы можем производить больше за меньшие деньги, мы создаем искусство, которое дополняет и даже превосходит реальность, мы расширили наш разум и усовершенствовали тела, и мы изучаем космос намного глубже и шире, чем когда-либо могли себе это представить.

Более того, благодаря тому что компьютеры стали дешевыми, почти каждый может научиться использовать их для создания вещей, которые ограничены только нашим воображением. Вот тут-то и появляется *вычислительное мышление* — способ рассмотрения окружающего мира с точки зрения абстракций и алгоритмов, которые позволяют превратить плоды нашего воображения в системы, работающие на наших компьютерах. Для создания программы, которая будет вести учет ваших финансов, или управлять системой отопления вашего дома, или рассчитывать траекторию ракеты, запущенной в космос, вы будете мыслить главным образом символами. Но если вы хотите построить беспилотный автомобиль, или сконструировать робота, безопасно взаимодействующего с людьми, или видеоигру, в которой есть сложные виртуальные противники, вам придется взаимодействовать с компьютером уже по-другому, и именно в этот момент в игру вступают искусственный интеллект и машинное обучение. Еще со времен Алана Тьюринга, одного из пионеров исследований в области вычислительной техники, ученые пытались заставить компьютеры рассуждать, учиться и вести себя как люди. Последние же достижения в области машинного обучения позволяют нам это делать уже сегодня.

Прочитав эту книгу, вы изучите основы искусственного интеллекта и машинного обучения в достаточной степени для того, чтобы создавать свои собственные игры и умных помощников, а также научиться писать программы, способные распознавать изображения и стиль текста.

Больше всего меня восхищает в этой книге то, что она абсолютно практико-ориентированная, поскольку поможет вам создавать системы искусственного интеллекта, которые очень важны для современной компьютерной науки. Кроме того, эта книга затрагивает сложные и спорные моменты, касающиеся предубеждений и различного рода этических вопросов использования искусственного интеллекта и машинного обучения, и эти моменты будут становиться все более и более важными по мере того, как вы будете продвигаться в своем обучении.

Я бы хотел, чтобы в детстве у меня была такая книга. Но в то время ее просто не могло быть, потому что большинство описанных в ней вещей еще не были изобретены. Поэтому вот вам мой вызов: узнав больше об искусственном интеллекте и машинном обучении, что нового изобретете вы?

*Гради Буч,
руководитель отдела проектирования программного
обеспечения исследовательского центра IBM*

Вам может показаться удивительным, но с системами искусственного интеллекта и машинного обучения сталкивался каждый. Большинство из нас используют системы искусственного интеллекта каждый день. Они оказывают влияние на новости, которые мы читаем и слышим, на решения, которые принимают компании и правительства, на выбор того, что мы будем покупать, смотреть и слушать. Они могут даже повлиять на то, кем мы будем работать и где будем жить.

Отличный способ узнать больше об искусственном интеллекте — попробовать самим создать вещи, которые его используют. И книга, которую вы держите в руках, будет тому практическим руководством. Создав собственный проект искусственного интеллекта, вы на деле поймете, как он работает и на что способен, а также узнаете о рисках, связанных с использованием искусственного интеллекта, когда сами увидите, как что-то может пойти не так.

Выполнение собственных проектов, в основе которых лежат реальные способы использования искусственного интеллекта в нашем мире, также поможет вам разобраться, как именно системы искусственного интеллекта могут влиять на каждого из нас. Когда вы узнаете, как создаются приложения, использующие искусственный интеллект, вы начнете чаще замечать его вокруг себя. Это также позволит вам лучше понять устройство окружающего мира.

Искусственный интеллект часто упоминается в новостях, но вам иногда может быть трудно понять их смысл, если нет базового понимания технологий, задействованных при его создании. Проекты из этой книги также подготовят вас к обсуждению различных вопросов использования искусственного интеллекта, а также его контроля и регулирования.

А еще проекты в этой книге развлекательные! Машинное обучение — это крайне увлекательная область технологий, которая позволяет создавать абсолютно уникальные вещи. Я надеюсь, что вам понравится придумывать и воплощать в реальность что-то новое, о чем вы и не знали раньше, до знакомства с этим инструментом.

Scratch



Главы этой книги познакомят вас с разными идеями применения машинного обучения через выполнение практического проекта в образовательной среде визуального программирования Scratch. Возможно, вы уже познакомились с этой средой, работая в ней на уроках в школе или на дополнительных курсах.

Если же вы никогда не слышали о Scratch, не переживайте — все проекты в книге сопровождаются подробными пошаговыми инструкциями. Будет лучше для начала посетить официальный сайт (<https://scratch.mit.edu/>), чтобы ознакомиться с возможностями этой среды.

И прежде чем начать выполнение проектов, ознакомьтесь с основной терминологией Scratch, которая будет часто использоваться в этой книге.

Знакомство с интерфейсом Scratch

Основные рабочие области, из которых состоит интерфейс среды Scratch, обозначены на рисунке 1 цифрами в кружочках.

1. *Главное меню* содержит различные опции для создания вашего проекта, его сохранения или загрузки из файла, а также ссылку на библиотеку шаблонов проектов, содержащую набор проектов с различными начальными настройками, позволяющих сэкономить время.

2. *Панель инструментов* содержит набор блоков, доступных для использования в вашей программе.

3. *Область кода* — это рабочая область, в которой вы создаете свои программы, перетаскивая в нее блоки из Панели инструментов. Она доступна, когда выбрана вкладка «Код». Если же выбрана вкладка «Фоны» или «Костюмы», Область кода содержит различные инструменты для рисования и называется *холстом*.

4. *Элементы управления* позволяют вам запускать вашу программу, когда вы захотите ее протестировать, а также управлять ходом ее выполнения и режимом отображения. Зеленый флажок запускает программу, а красный кружочек рядом останавливает ее выполнение. Крайняя правая кнопка с четырьмя стрелками запускает вашу программу в полноэкранном режиме.

5. *Сцена* — рабочая область, в которой отображается визуальная часть (интерфейс) вашей программы. Именно в этой области будут взаимодействовать различные компоненты вашего проекта.

6. *Спрайты* — это объекты, которые выполняют различные действия в вашем проекте. Каждый спрайт имеет характерный

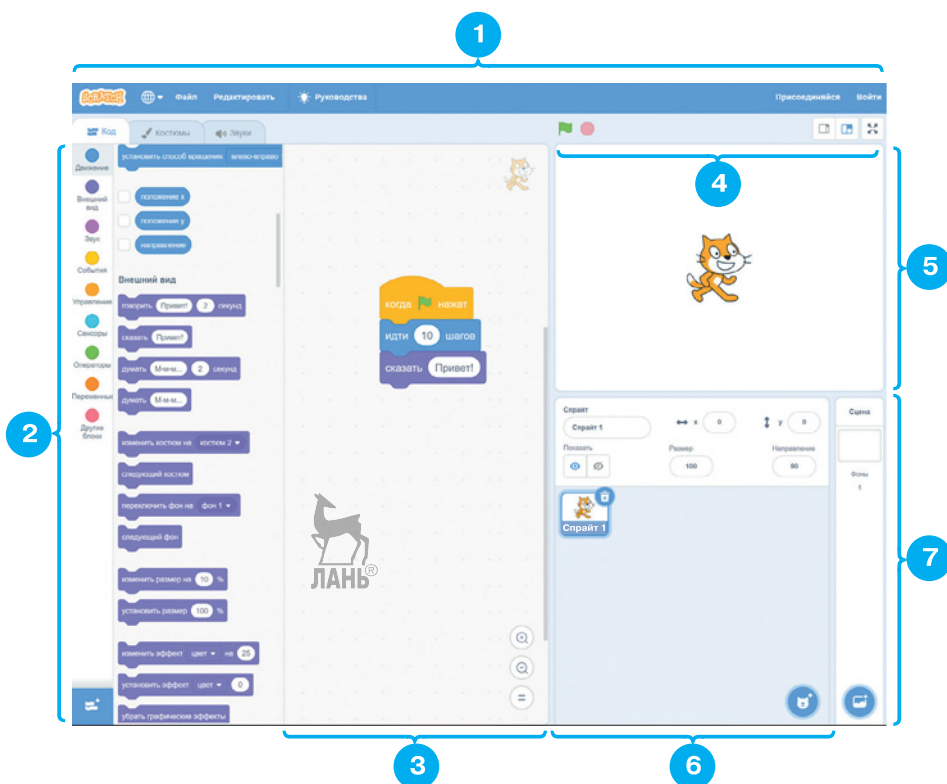


Рис. 1. Интерфейс среды Scratch 3

внешний вид (называются *костюмы*) и свой собственный код в Области кода.

7. *Фоны* — рабочая область вашего проекта. Вы можете выбирать готовые варианты из стандартной библиотеки фонов или создавать свои собственные.

Программирование в среде Scratch

Для создания программ (в Scratch они называются *скрипты*) вам нужно перетаскивать блоки из Панели инструментов в Область кода. Когда вы приближаете их достаточно близко друг к другу, они соединяются.

Цвета и категории блоков

Блоки в Scratch объединены в категории, каждая из которых имеет свой цвет. Например, в категории «Движение» располо-

жены блоки, предназначенные для управления перемещением спрайтов, и все они синего цвета. Для того чтобы увидеть их, нажмите на синий кружок с подписью «Движение» на Панели инструментов.

Когда вы будете воспроизводить примеры из этой книги, подобная цветовая разметка поможет вам быстрее и проще находить нужные блоки. Например, если в проекте требуется блок желтого цвета, нажмите на желтый кружок с подписью «События» на Панели инструментов, и вы сразу же перейдете ко всем блокам категории «События».

Создание пользовательских блоков

Категория розового цвета «Другие блоки» нужна для создания и хранения ваших собственных блоков — пользовательских блоков. Такие блоки нужны для того, чтобы разбить длинный скрипт на более мелкие части, чтобы его было легче читать и чтобы в нем было легче ориентироваться.

Для создания пользовательского блока нажмите на кнопку «Создать блок» в категории «Другие блоки» на Панели инструментов, как показано на рисунке 2.

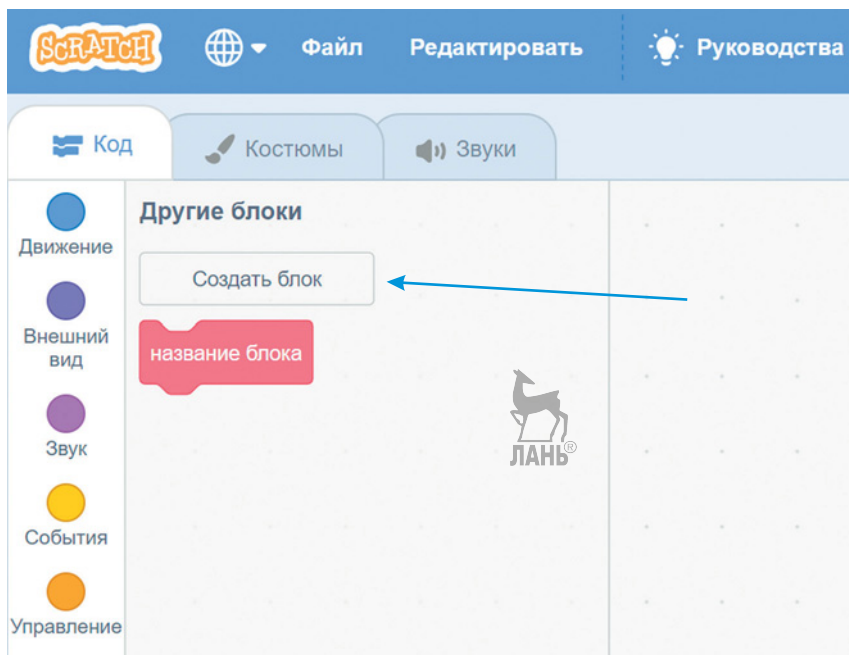


Рис. 2. Нажмите на кнопку «Создать блок» для создания пользовательского блока

Придумайте вашему пользовательскому блоку имя, затем нажмите на кнопку «ОК». Будет создан новый блок розового цвета, который вы сможете использовать так же, как и все остальные блоки Scratch.

Разместите часть скрипта со всеми шагами, которые по вашему мнению должны выполняться в пользовательском блоке, под большим розовым блоком «Определить», как показано на рисунке 3.

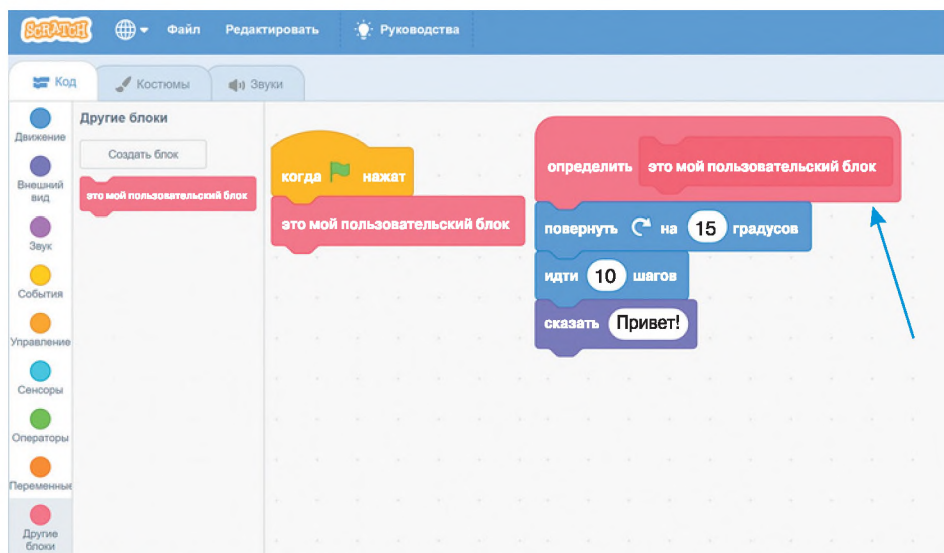


Рис. 3. Пользовательские блоки доступны на Панели инструментов и могут быть использованы так же, как и все остальные блоки Scratch

Дублирование кода

Если вы создаете достаточно длинный скрипт, в нем могут встречаться повторяющиеся фрагменты. Для экономии времени вы можете их дублировать.

Щелкните правой кнопкой мыши по фрагменту скрипта, который хотите дублировать, и в открывшемся контекстном меню выберите команду «Дублировать». Эта функция может оказаться очень полезной при выполнении проектов этой книги.

Сохранение результатов вашей работы

Очень важно периодически сохранять проект во время создания скриптов в Scratch, поскольку Scratch не делает этого автоматически.

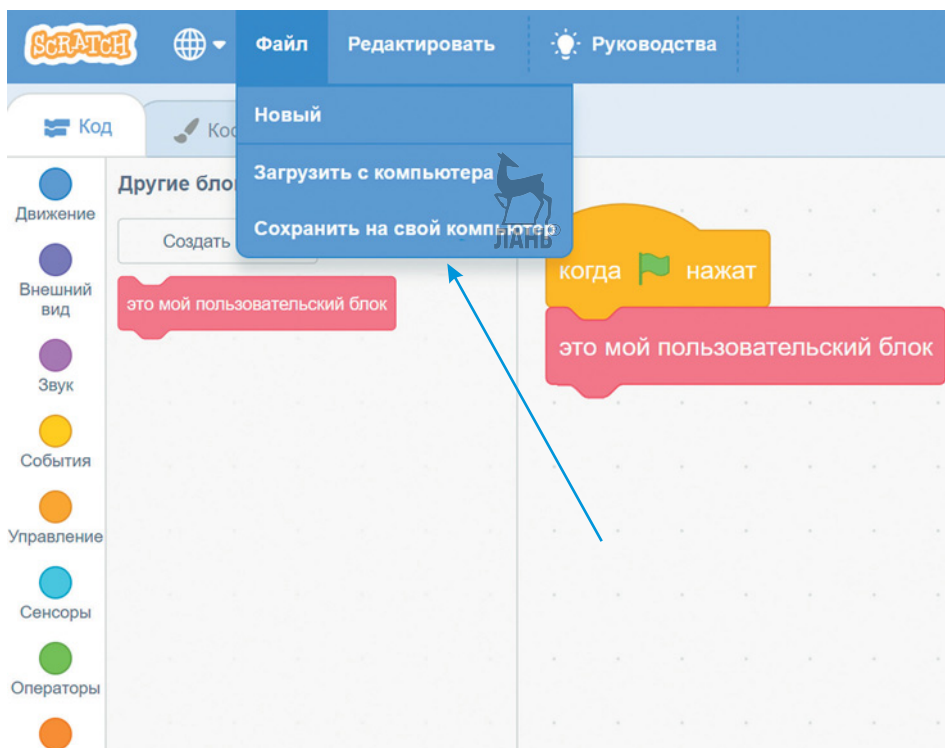


Рис. 4. Меню **Файл** позволяет вам сохранить проект и продолжить работу над ним позже

Чтобы сохранить проект, в Главном меню выберите **Файл** → **Сохранить на свой компьютер**, как показано на рисунке 4. После этого на ваш компьютер будет скачан файл. Сохраните этот файл — в нем копия созданного вами проекта.

Чтобы снова вернуться к работе над этим проектом, нажмите **Файл** → **Загрузить с компьютера** и выберите файл, который вы скачали ранее.



Проект «Машинное обучение для детей»

При выполнении большинства проектов этой книги вы будете использовать бесплатный онлайн-инструмент «Машинное обучение для детей», который расширяет функционал среды Scratch, добавляя ей возможность создания проектов по машинному обучению. Подробные инструкции по началу работы с этим инструментом вы найдете в главе 2.

Что дальше?

Содержание этой книги выстроено следующим образом:

Глава 1. Что такое искусственный интеллект?

Вы узнаете больше об искусственном интеллекте и машинном обучении, а также о том, почему в проектах этой книги используется машинное обучение вместо традиционного программирования.

Глава 2. Знакомство с онлайн-инструментом «Машинное обучение для детей»

В этой главе вы познакомитесь с инструментом, который будете использовать при выполнении большинства проектов по машинному обучению.

Остальная часть книги посвящена различным вещам, распознаванию которых можно обучить системы машинного обучения.

Глава 3. Сортировка изображений животных

В этой главе вы узнаете больше о распознавании изображений. Вы обучите компьютер распознавать объекты на фотографии, а затем автоматически сортировать изображения животных.

Глава 4. Игра в «Камень, ножницы, бумага» с компьютером

В проекте этой главы вы будете использовать веб-камеру, чтобы обучить систему машинного обучения распознавать разные формы рук. В результате вы сможете играть в «Камень, ножницы, бумага» со своим компьютером!

Глава 5. Распознавание постеров фильмов

Вы узнаете, как можно научить компьютер распознавать не только объекты на изображениях, но и художественные стили. Обучив компьютер «судить о книге по обложке», вы узнаете, есть ли у него способности к творчеству.

Глава 6. Сортировка писем

В этой главе вы узнаете, как обучить компьютер распознавать рукописные тексты. Затем вы создадите простую систему для сортировки писем на основе распознавания почерка.

Глава 7. Распознавание эмоций

В этой главе вы узнаете, как обучить компьютер определять эмоциональную окраску письма. Вы создадите игру и обучите ее персонажа распознавать ваши комплименты и оскорбления и даже реагировать на них.

Глава 8. Распознавание стиля письма в газетных статьях

Вы обучите компьютер распознавать различные стили письма, чтобы определять, в какой газете могла выйти рассматриваемая статья. Вы также узнаете об основных способах оценки качества систем машинного обучения.

Глава 9. Поиск объекта на картинке

Проект этой главы основан на выполненных вами ранее проектах и состоит в обучении компьютера находить наименьший объект на изображении. Вы также узнаете о некоторых способах применения этой технологии в реальных приложениях для таких сфер, как обработка спутниковых изображений или обучение беспилотных автомобилей.

Глава 10. Умные помощники

В этой главе вы узнаете, как обучить компьютер понимать смысл текста и как эта технология используется при программировании умных помощников. Затем вы создадите себе простого умного помощника, который будет понимать ваши команды и включать-выключать разные устройства.

Глава 11. Чатботы

Вы узнаете больше о чатботах и о том, как создать систему ответов на вопросы, если обучить компьютер понимать смысл текста.

Глава 12. Создание игры «Убеги от монстра»

В этой главе вы узнаете, какую роль в развитии технологии искусственного интеллекта сыграли компьютерные игры. Вы обучите систему машинного обучения играть в упрощенную версию легендарной игры «Рас-Ман»^{*}.

Глава 13. Крестики-нолики

Вы познакомитесь с еще одним примером взаимодействия компьютерных игр и искусственного интеллекта и воспроизведете версию известного исследовательского проекта в области искусственного интеллекта по обучению компьютера игре «Крестики-нолики».

Глава 14. Обработка ошибок

Вы сами убедитесь в том, что в проектах машинного обучения чаще всего что-то идет не так. Для этого вы создадите собственную достаточно запутанную систему, в которой есть ошибки. Узнаете о связанных с этим проблемах, и о шагах, которые можно предпринять, чтобы этих проблем избежать.

^{*} Культовая компьютерная игра в жанре аркады, впервые вышедшая в 1979 году в Японии. — Прим. ред.



Глава 15. Этические вопросы использования искусственного интеллекта

В этой главе вы узнаете, как некоторые разработчики намеренно влияют на ответы, которые дают их системы машинного обучения, а также о некоторых этических вопросах использования искусственного интеллекта, возникающих в связи с этим.

Послесловие

В конце книги мы поразмышляем, каким же может оказаться будущее искусственного интеллекта.





ГЛАВА 1

Что такое искусственный интеллект?

https://t.me/it_boooks/2

В этой книге мы будем изучать различные аспекты использования искусственного интеллекта, выполняя проекты с применением машинного обучения для решения задач реального уровня сложности. Однако, прежде чем мы начнем, было бы полезно ознакомиться с некоторыми справочными сведениями об инструментах и технологиях, с которыми мы будем работать.

Во введении вы познакомились с языком программирования Scratch и узнали о функционале каждого раздела интерфейса среды Scratch. В этой главе вы найдете определения и разъяснения основных терминов и концепций программирования, которые будут использоваться в остальных главах этой книги.

Программирование

Программирование — это способ объяснения машинам того, что мы хотим, чтобы они сделали. Чаще всего мы используем программный код для управления компьютерами, но он также используется для управления небольшими устройствами (такими как мобильные телефоны), бытовыми приборами (такими как стиральные машины) и огромными машинами (автомобилями и самолетами).

Чтобы написать код, сначала нужно определиться с задачей, которую мы хотим, чтобы машина выполняла, а затем разбить эту задачу на ряд команд. Команды должны быть конкретными и достаточно подробными, чтобы машина могла им следовать.

Некоторые языки программирования, такие как Scratch, представляют программирование с помощью цветных визуальных блоков, обозначающих эти шаги. Соединяя блоки вместе, вы тем самым описываете последовательность шагов, которые должна выполнить машина (рис. 1.1).

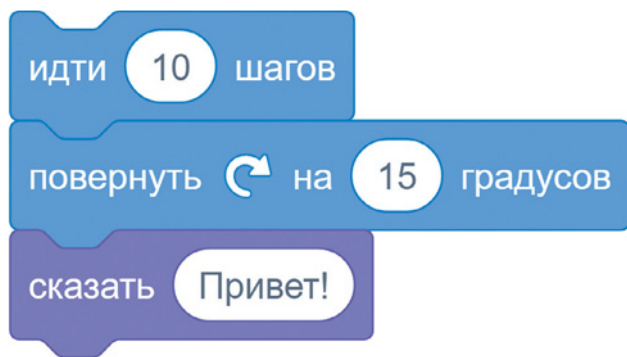


Рис. 1.1. Scratch представляет программирование с помощью соединенных между собой цветных блоков

Описание задачи как последовательности шагов, которые должна выполнить машина, было задачей разработчиков программного обеспечения многие десятилетия. Теперь существуют более интуитивно понятные языки программирования, которые значительно упрощают описание шагов, но основная идея все еще осталась прежней.

Тем не менее на сегодняшний день машины стали настолько сложными, что одного программирования уже недостаточно, чтобы управлять ими. Например, выполнение основных задач современного автомобиля описывается более чем 100 миллионами строк кода. Это все равно что описать 100 миллионов шагов!

Некоторые задачи, которые мы ставим перед современными машинами, настолько сложны, что написание пошаговых инструкций заняло бы слишком много времени. А иногда мы даже не до конца понимаем, как описать нужные шаги.

Для решения задач такого рода используется машинное обучение.

Машинное обучение

Машинное обучение полезно, когда мы имеем дело с задачами, написание последовательных шагов (подробных инструкций) для решения которых требует слишком много времени.

Тогда вместо описания конкретных команд, которые должна будет выполнять машина, мы будем предлагать ей примеры решения подобных задач до тех пор, пока она не научится решать их самостоятельно.

Представьте, что кто-то учится бить по мячу. Вы можете давать ему четкие пошаговые инструкции: как высоко поднимать ногу перед ударом, как быстро ею двигать, что в этот момент делать с руками и так далее. Такое представление пошаговых инструкций есть традиционный подход к программированию.

В то же время вы можете показать этому человеку множество примеров того, как разные люди бьют по мячам разных видов, и таким образом научить его. В этом и состоит основной принцип машинного обучения — обучение через подбор и демонстрацию большого количества примеров решения выполняемой задачи.

В этой книге вы найдете множество примеров того, как обучаются системы машинного обучения, как они себя ведут и как используются в окружающем мире.

Искусственный интеллект

Люди иногда путают *машинное обучение* с *искусственным интеллектом*. Некоторые наиболее интересные системы искусственного интеллекта были созданы при использовании машинного обучения, но на самом деле это далеко не единственный способ создания таких систем. Взаимосвязь между машинным обучением и искусственным интеллектом показана на рисунке 1.2.

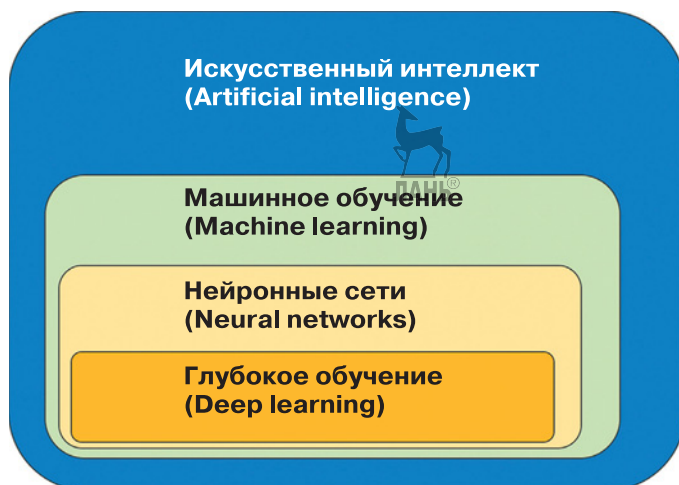


Рис. 1.2. Взаимосвязь между машинным обучением и искусственным интеллектом



Искусственный интеллект — это более общий термин. Им называют проекты, в которых машины выполняют то, что обычно требует участия человека (человеческого интеллекта). Но термин «искусственный интеллект» не подразумевает конкретного способа, как именно заставить машину это делать. А таких способов много, и машинное обучение всего лишь один из них. Так же как и нейронные сети, о которых мы поговорим совсем скоро, это лишь один из видов машинного обучения. Но чтобы лучше понять эту разницу, давайте рассмотрим пример.

В 1997 году шахматный суперкомпьютер Deep Blue обыграл чемпиона мира по шахматам Гарри Каспарова. Многие считали Deep Blue важной вехой в развитии искусственного интеллекта.

Но Deep Blue не был системой машинного обучения. Он не научился сам играть в шахматы, он был только лишь запрограммированным устройством. Разработчики закодировали и внесли в его систему правила игры в шахматы и, главное, выигрышные стратегии. То есть в ходе игры компьютер просто выполнял пошаговые инструкции. Deep Blue не был умнее Каспарова, но он мог одновременно просчитывать гораздо больше возможных ходов. Этого оказалось достаточно для победы.

В 2011 году суперкомпьютер IBM Watson принял участие в американской телевизионной викторине «Jeopardy!», обыграв чемпионов этой игры Брэда Раттера и Кена Дженнингса. IBM Watson также разрабатывался как проект искусственного интеллекта, и он продемонстрировал потенциал компьютерных систем в области распознавания естественного языка. Однако в отличие от Deep Blue IBM Watson был системой машинного обучения. Он обучился играть в «Jeopardy!», отвечая на вопросы из каждой серии этой телевикторины начиная с 1960-х годов, участвуя во множестве тренировочных матчей с соперниками-людьми.

Тем не менее такие системы искусственного интеллекта, как Deep Blue, все еще создаются сегодня, потому что простые системы, которые следуют пошаговым инструкциям, могут быть все так же полезны. И все же для создания систем искусственного интеллекта, способных решать более сложные задачи, используется машинное обучение.

Нейронные сети и глубокое обучение

Нейронные сети и глубокое обучение (см. рис. 1.2) — это два типа машинного обучения. В этой книге они не будут рассматриваться подробно, но, поскольку они часто упоминаются в но-

востях и статьях об искусственном интеллекте, давайте выясним, как они связаны друг с другом.

Нейронные сети — это популярный и очень эффективный способ создания систем машинного обучения. Доказана их эффективность даже при решении особо сложных задач. В общих чертах, структура нейронных сетей формируется по образцу структуры мозга животных. При этом различные части системы машинного обучения (нейроны) располагаются слоями, которые связаны друг с другом.

Глубокое обучение — это метод работы с нейронной сетью, состоящей из большого количества слоев. На сегодняшний день это один из самых эффективных методов, используемых при создании систем искусственного интеллекта.

В этой книге основное внимание будет уделяться машинному обучению в целом, без досконального разъяснения различных подходов к созданию систем машинного обучения. Вы узнаете, как ведут себя системы машинного обучения, как они обучаются, как используются в реальном мире, а также какие проблемы могут с ними возникнуть. Если же вы заинтересуетесь конкретными инструментами машинного обучения, такими как нейронные сети и глубокое обучение, то после прочтения книги у вас будут все необходимые для этого начальные знания.

Что вы узнали

Машинное обучение — обучение компьютера решению задачи путем сбора большого количества примеров решения этой задачи. Это популярный на сегодняшний день способ реализации проектов в области искусственного интеллекта, потому что он позволяет обучить компьютер решению более сложных задач, чем те, решение которых может быть описано пошаговыми инструкциями.





ГЛАВА 2

Знакомство с онлайн-инструментом «Машинное обучение для детей»

В этой книге вы будете выполнять проекты по машинному обучению с использованием бесплатного образовательного онлайн-инструмента под названием «Машинное обучение для детей». Вы узнаете, как работает этот инструмент, как использовать его в проектах и как ваши родители или учитель могут настроить его для вас.

Для доступа к нему перейдите по ссылке (<https://MachineLearningForKids.co.uk/>) и, чтобы каждый раз не вводить вручную такой длинный адрес, просто сохраните его в закладки.

На рисунке 2.1 показана главная страница сайта*. Впрочем, вы наверняка знаете, что дизайн всех сайтов со временем меняется,

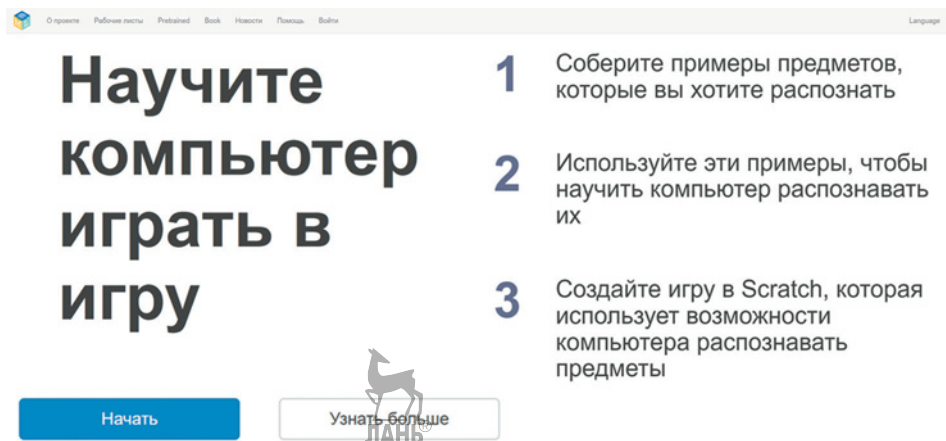


Рис. 2.1. Главная страница сайта «Машинное обучение для детей»

* Поскольку этот сайт англоязычный, в первую очередь нажмите на кнопку «Language» (в правом верхнем углу страницы) и в появившемся списке выберите русский язык. Следует отметить, однако, что русская локализация у сайта часто обновляется. Она создана с помощью автоматического перевода, поэтому названия на скриншотах в книге могут немного различаться. — *Прим. перев.*

поэтому внешний вид главной страницы может несколько отличаться от того, что вы видите на рисунке.

Вход в систему

Для выполнения каждого из проектов этой книги вам нужно будет в первую очередь заходить на сайт «Машинное обучение для детей» и выполнять вход в систему.

Выберите команду «Войти» в Главном меню сайта (в верхней части главной страницы). Отобразится страница авторизации (рис. 2.2).

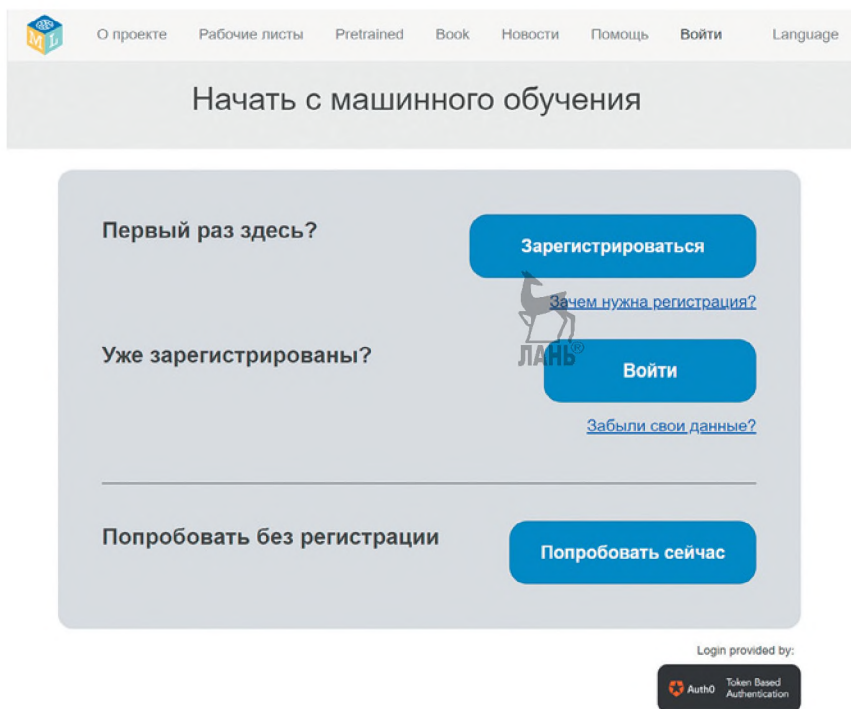


Рис. 2.2. Страница авторизации сайта «Машинное обучение для детей»

На этой странице вам нужно выбрать один из вариантов:

- Кнопка «Зарегистрироваться». Нажатие на эту кнопку позволит создать новый аккаунт (это бесплатно). Попросите кого-нибудь из родителей или учителей помочь вам с этим. Подробные инструкции, как создать аккаунт, приведены в этой главе в разделе «Создание аккаунта» на с. 35.

- Кнопка «Войти». Если кто-то из ваших родителей или учителей уже создал для вас новый аккаунт (это бесплатно), нажмите на кнопку «Войти» и во всплывающее окно введите свои имя пользователя (username) и пароль (password). После входа в свой аккаунт вы сможете сохранять проекты и возвращаться к ним в любой момент, чтобы продолжить работу, с любого устройства: домашнего компьютера, любимого планшета или школьного рабочего места.
- Кнопка «Попробовать сейчас». Нажмите эту кнопку, если у вас еще нет своего аккаунта. Без регистрации вы все равно сможете создать проект, но он будет доступен только в течение четырех часов. Этого может быть достаточно для выполнения любого проекта из этой книги, но вы не сможете вернуться к нему позднее и показать кому-то.

Создание нового проекта

После входа в систему вы попадаете на страницу со списком ваших проектов (рис. 2.3). Вы можете в любой момент возвращаться к этой странице, выбрав раздел «Проекты» в главном меню сайта.

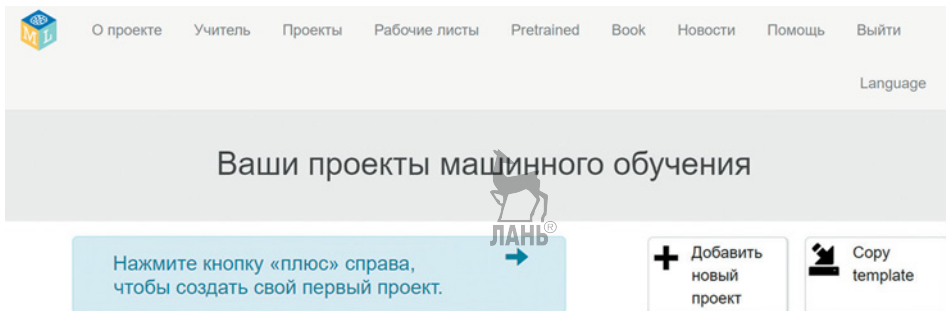


Рис. 2.3. Страница со списком ваших проектов (пустая, если вы еще не создали ни одного проекта)

Для создания нового проекта выполните следующие шаги:

1. Нажмите на кнопку «Добавить новый проект» (см. рис. 2.3).
2. Придумайте имя своему проекту и введите его в поле «Название проекта», как показано на рисунке 2.4. В каждой главе этой книги вам будет предлагаться название, но вы можете вводить и собственные, если захотите. Фантазируйте!

О проекте Учитель Проекты Рабочие листы Pretrained Book Новости Помощь Выйти

Language

Начать новый проект машинного обучения

☐ Проект всего класса?

Название проекта *

Распознавание *

СОЗДАТЬ ОТМЕНА

ЛАНЬ®

Поставьте галочку, если вы хотите, чтобы весь ваш класс мог работать над этим проектом совместно.

Это полезно для проектов, которые преподают краудсорсинг в качестве подхода к обучению проектов машинного обучения.

Рис. 2.4. Создание нового проекта машинного обучения

О проекте Учитель Проекты Рабочие листы Pretrained Book Новости Помощь Выйти

Language

Начать новый проект машинного обучения

☐ Проект всего класса?

Название проекта

Мой проект

текст

изображения

числа

звуки

СОЗДАТЬ ОТМЕНА

ЛАНЬ®

Какие типы данных вы хотите научить компьютер распознавать?

Для слов, предложений или абзацев выберите «текст»

Для фотографий, диаграмм и картинок выберите «изображения»

Для набора чисел или множественного выбора выберите «числа»

Для голосов и звуков выберите «звуки»

Рис. 2.5. Выбор типа данных для проекта

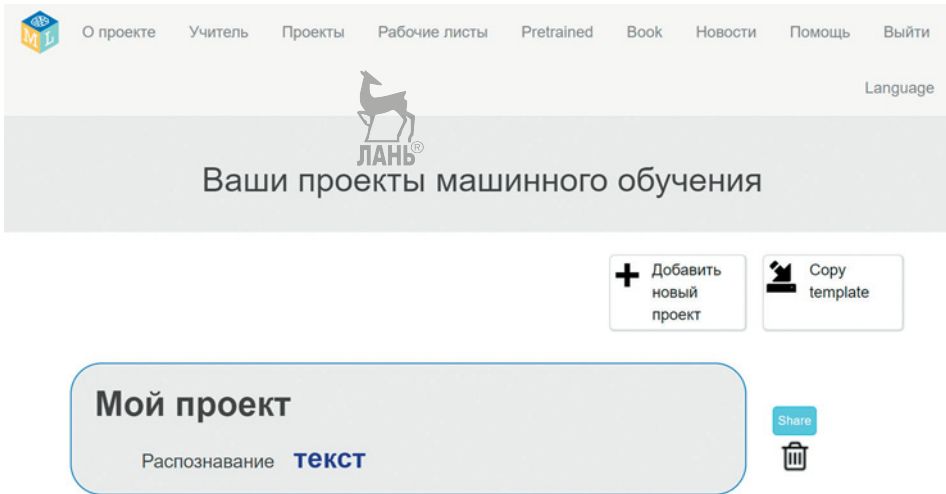


Рис. 2.6. Обновленный список проектов

3. Нажмите на пункт «Распознавание» (см. рис. 2.4). Откроется выпадающий список, в котором перечислены разные типы проектов машинного обучения, как показано на рисунке 2.5. Здесь вы будете выбирать тип входящих данных (например, текст или изображения), который компьютер должен научиться распознавать в вашем проекте. В каждом проекте этой книги будет подсказка, какой тип данных нужно выбрать. Вы не запутаетесь.

4. Нажмите на кнопку «Создать».

5. Вы вернетесь к списку ваших проектов (рис. 2.6). Нажмите на название проекта, который вы только что создали, чтобы начать.

Этапы работы над проектом

Работа над каждым из ваших проектов машинного обучения будет состоять из трех основных этапов: «Обучить», «Узнать и проверить» и «Создать», как показано на рисунке 2.7.

Вы можете переходить от одного этапа к другому, нажимая на синие кнопки под их названиями. В какой момент это нужно будет делать, подскажут инструкции к выполнению каждого из проектов.

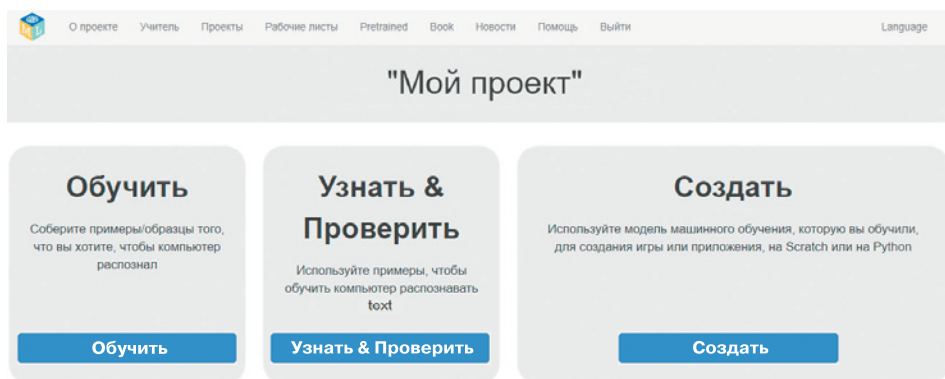


Рис. 2.7. Этапы работы над проектом машинного обучения

Этап «Обучить»

На этом этапе вы собираете примеры того, что должен обучиться распознавать компьютер.

Вы будете создавать коллекцию* для каждой подкатегории объектов, которые компьютер должен научиться распознавать и отличать друг от друга. Каждая коллекция будет находиться внутри серой рамки, как показано на рисунке 2.8.

После этого вы будете наполнять коллекции примерами. Если при создании проекта вы выбрали тип данных «текст», то каждую из коллекций вы должны наполнить примерами текстов, которые можно отнести к каждой из подкатегорий. На рисунке 2.8 показаны две коллекции, наполненные текстовыми примерами для подкатегорий «something» и «something else»**. Например, в проекте из главы 7 этой книги вы будете наполнять коллекции для подкатегорий «kind» и «mean»***, заполняя их примерами комплиментов и оскорблений. Для проектов с типом данных «изображение» вы будете наполнять коллекции примерами изображений, соответствующих каждой из подкатегорий. В проектах с типом данных «звук» коллекции нужно будет наполнять примерами звукозаписей.

В правом нижнем углу каждой коллекции отображается число. Это счетчик, который показывает, сколько примеров вы уже добавили в коллекцию.

* При создании новой коллекции обратите внимание, что название коллекции может содержать только буквы латиницы и цифры. — Прим. ред.

** «Something» переводится с английского как «что-то», «something else» — как «что-то другое». — Прим. пер.

*** Слова «kind» переводится с английского как «добрый», «mean» — как «подлый». — Прим. пер.

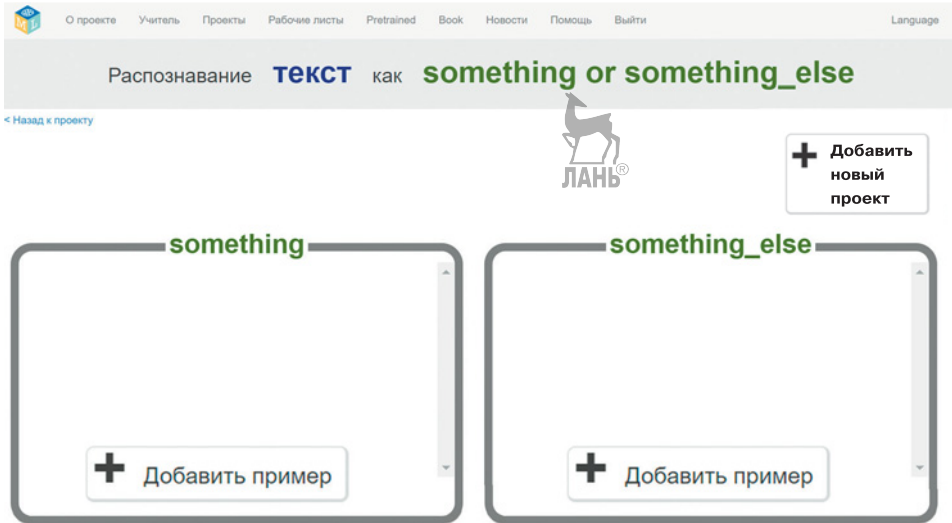


Рис. 2.8. Этап «Обучить»

Если вы добавили пример в коллекцию случайно и хотите его удалить, наведите на него указатель мыши и нажмите на появившийся красный крестик. Не бойтесь ошибаться!

Если же вы хотите удалить всю коллекцию, вместе со всеми примерами в ней, наведите указатель мыши на правый верхний угол коллекции. В нем также появится красный крестик, нажатие на который приведет к удалению всей коллекции. Будьте внимательны: восстановить коллекцию после удаления уже нельзя!

Этап «Узнать и проверить»

Как только вы соберете достаточно большое количество примеров, их можно будет использовать для обучения модели машинного обучения. Более подробно о таких моделях вы узнаете в следующей главе.

Чтобы начать процесс обучения, нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 2.9).

Насколько долгим будет процесс обучения, зависит от выбранного типа данных и от количества собранных вами примеров. Например, проекты с типом данных «изображение» обучаются дольше, чем проекты с типом данных «текст», поскольку компьютеру гораздо труднее распознать изображение, чем текст или числа. При этом, чем больше примеров вы собрали, тем больше времени займет обучение. А иногда обучение длится дольше, если сервер окажется занят.

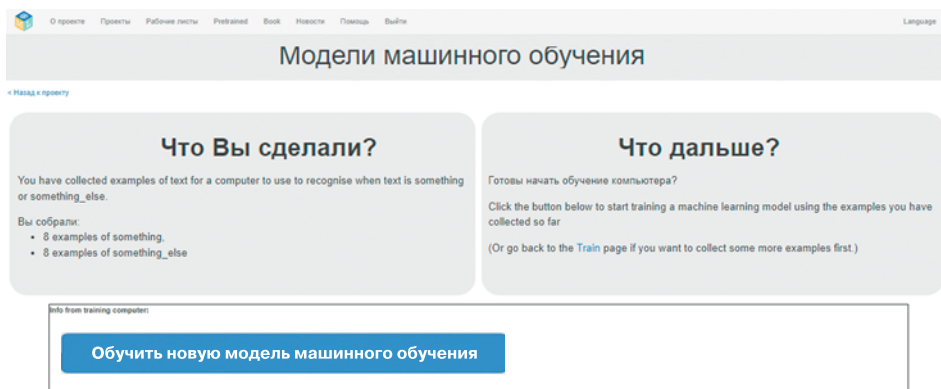


Рис. 2.9. Обучение модели

Процесс обучения может занять 30 секунд, а может и несколько минут. Наберитесь терпения! К тому же, чтобы вы не скучали, внизу страницы отображается викторина.

Но если вы собрали слишком мало примеров, кнопка «Обучить новую модель машинного обучения» не отобразится. Вернитесь к этапу «Обучить» и добавьте больше примеров в коллекцию.

Этап «Создать»



После того как вы обучили модель машинного обучения, вы можете ее использовать для создания чего-то интересного.

«Машинное обучение для детей» предлагает разные варианты проектов, в которых можно использовать вашу модель, но все проекты в этой книге будут выполняться с использованием Scratch 3. Поэтому для продолжения работы над проектами на этапе «Создать» вам нужно будет нажимать на кнопку «Scratch 3» (рис. 2.10).

Теперь вы готовы приступить к выполнению проектов из этой книги!

Но помните: если вы хотите иметь возможность сохранять свои проекты и выбирать вариант «Войти», а не только «Попробовать сейчас», вам нужно будет попросить кого-нибудь из взрослых создать для вас аккаунт. Далее приведены инструкции, как это сделать.

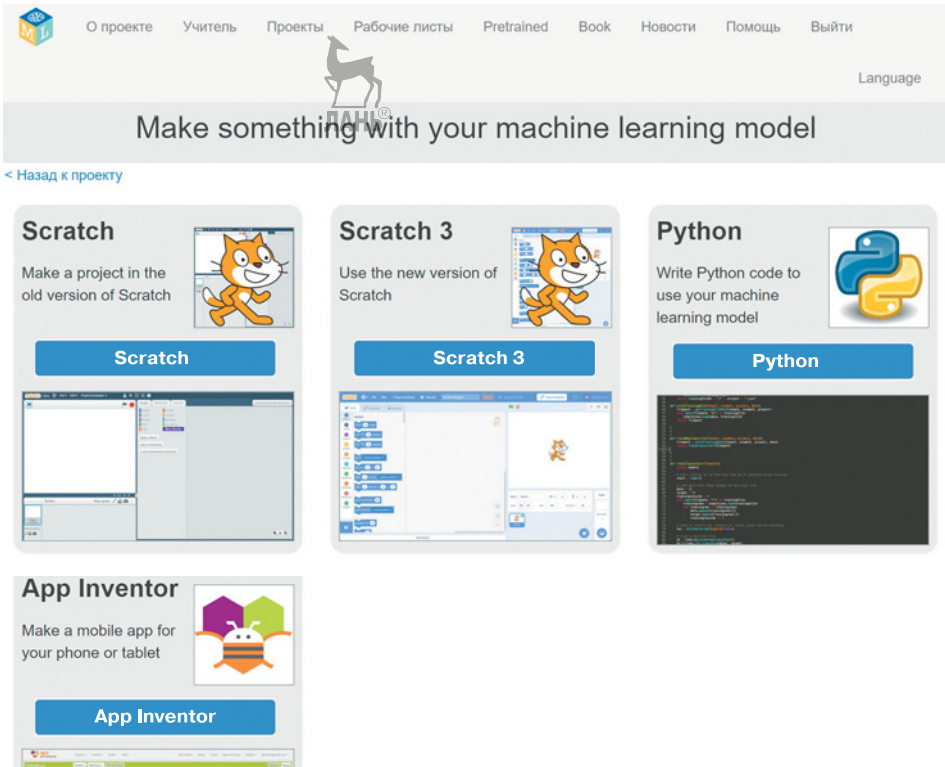


Рис. 2.10. Создание проекта машинного обучения

Создание аккаунта

Аккаунт создается только один раз, и это совершенно бесплатно. Создание аккаунта состоит из 10 шагов и занимает около 10 минут. Вашему родителю или учителю нужно лишь выполнить приведенные ниже инструкции.



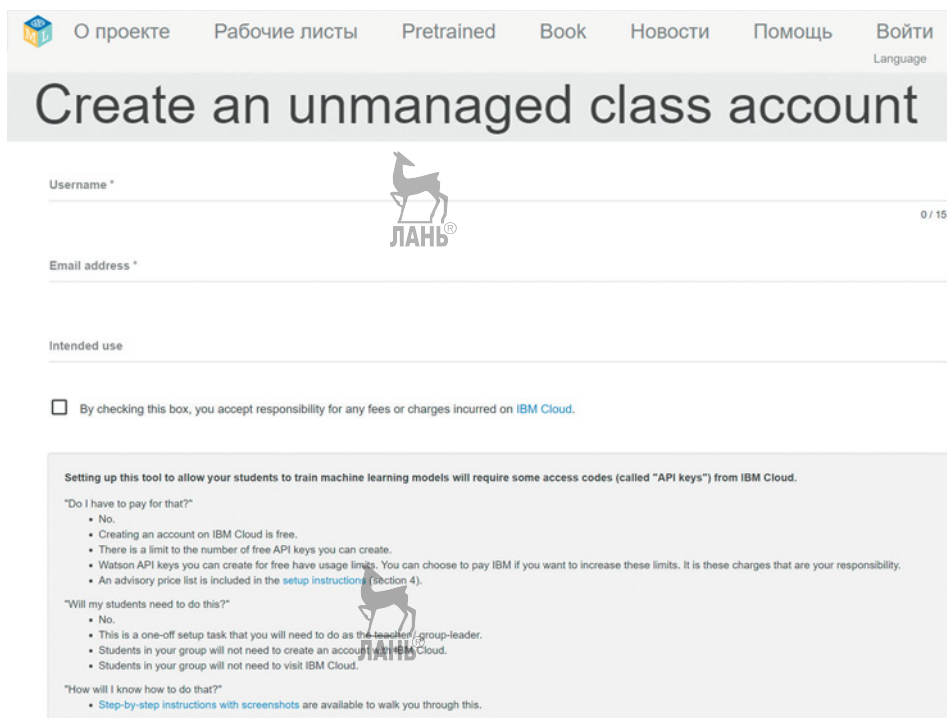
Примечание

При создании аккаунта нужно будет указать адрес электронной почты. С подробной информацией о том, как этот адрес будет использоваться, можно ознакомиться по ссылке <https://machinelearningforkids.co.uk/help/>

1. На странице входа нажмите на кнопку «Зарегистрироваться» (см. рис. 2.2).
2. Нажмите на кнопку «Родитель, учитель или руководитель клуба программирования», чтобы подтвердить свой статус.

3. Нажмите на кнопку «Зарегистрироваться» в разделе «Создать неконтролируемую учетную запись класса». Это означает, что вы будете управлять своим аккаунтом и настраивать его самостоятельно.

4. Заполните форму, изображенную на рисунке 2.11. Выберите имя пользователя и введите действующий адрес электронной почты. Также дополнительно вы можете описать, для каких целей и каким образом планируете использовать данный сайт (поле необязательно для заполнения).



О проекте Рабочие листы Pretrained Book Новости Помощь Войти

Language

Create an unmanaged class account

Username *

Email address *

Intended use

☐ By checking this box, you accept responsibility for any fees or charges incurred on IBM Cloud.

Setting up this tool to allow your students to train machine learning models will require some access codes (called "API keys") from IBM Cloud.

"Do I have to pay for that?"

- No.
- Creating an account on IBM Cloud is free.
- There is a limit to the number of free API keys you can create.
- Watson API keys you can create for free have usage limits. You can choose to pay IBM if you want to increase these limits. It is these charges that are your responsibility.
- An advisory price list is included in the [setup instructions](#) (Section 4).

"Will my students need to do this?"

- No.
- This is a one-off setup task that you will need to do as the teacher/group-leader.
- Students in your group will not need to create an account with IBM Cloud.
- Students in your group will not need to visit IBM Cloud.

"How will I know how to do that?"

- [Step-by-step instructions with screenshots](#) are available to walk you through this.

Рис. 2.11. Создание аккаунта родителя или учителя

5. Вы получите электронное письмо для подтверждения указанного адреса электронной почты. Прежде чем перейти к следующему шагу, откройте это письмо и перейдите по содержащейся в нем ссылке.

6. После создания родительского/учительского аккаунта вы получите доступ к странице администрирования. Для перехода к ней в Главном меню сайта выберите опцию «Учитель».

7. Нажмите на кнопку «Ограничения», чтобы ознакомиться с ограничениями, которые по умолчанию установлены для вашего аккаунта. Некоторые из них вы можете изменить.

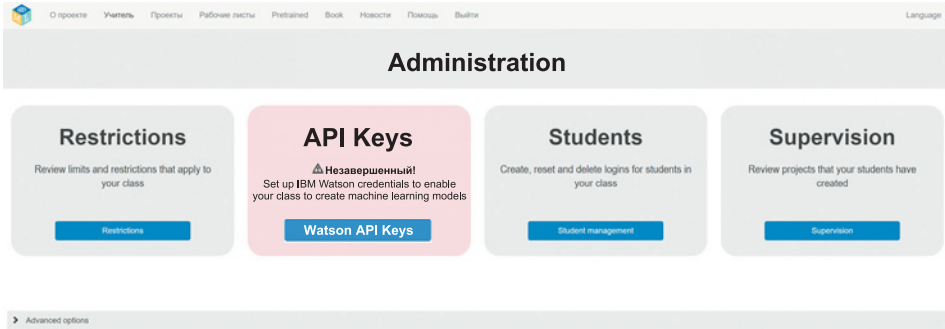


Рис. 2.12. Экран родительского/учительского аккаунта

8. Перейдите в раздел «API Keys» (API-ключи) из страницы администрирования (рис. 2.12), чтобы ввести коды от сервисов IBM Cloud, которые позволят использовать технологии машинного обучения в ваших проектах*. Вам понадобится как минимум один ключ для Watson Assistant.

Для получения таких кодов нужно создать бесплатный аккаунт IBM Cloud. Ключи для Watson Assistant также бесплатны.

Сайт IBM Cloud может показаться несколько сложным в использовании, если вы не привыкли пользоваться специализированными сайтами для разработчиков программного обеспечения. Поэтому на странице <https://machinelearningforkids.co.uk/apikeys-guide/> представлено пошаговое руководство по работе с ним.



Примечание

При создании API-ключа убедитесь, что выбрали категорию «Lite», так как именно она является бесплатной. Такие ключи несколько ограничивают возможности использования, но для выполнения проектов из этой книги их будет вполне достаточно.

9. Перейдите в раздел «Ученики», чтобы задать имя пользователя для ученика (ребенка), который будет выполнять проекты. Если проекты будут выполняться несколькими учениками, вы можете задать несколько имен пользователей.

Задавать реальные имена учеников или же предоставлять их контактные данные не нужно.

Создание аккаунта ученика гораздо проще, чем аккаунта родителя/учителя, так как ученикам не нужно вводить никаких ключей.

* Если вы открываете сайт с территории РФ, то можете пропустить этот шаг. Для проектов с распознаванием текста вы можете воспользоваться пунктом «Попробовать без регистрации» (см. рис. 2.2). — Прим. ред.

Если ученик забыл пароль, в родительском/учительском аккаунте может быть задан новый.

10. Перейдите в раздел «Наблюдение» для просмотра проектов, созданных учеником.

Вряд ли вы достигнете лимита на использование ключей API, добавляя новых учеников. Тем не менее это может случиться, если достаточно большое количество ваших учеников будет выполнять проекты одновременно. Если это произойдет, вы всегда сможете посмотреть, какие ключи используются в этой книге.

Что вы узнали



«Машинное обучение для детей» — это бесплатный онлайн-инструмент, который вы будете использовать при выполнении проектов из этой книги. Он проведет вас через основные этапы работы над проектом машинного обучения. Инструкции к выполнению проектов в книге подробно расскажут вам, что делать на каждом из этапов и когда переходить к следующему или возвращаться к предыдущему.

Если же вы хотите иметь возможность сохранять свои проекты, вам понадобится свой аккаунт. Попросите кого-нибудь из взрослых создать его. Создание аккаунта может занять около 10 минут, но это бесплатно, делается только один раз и сопровождается подробными инструкциями.





ГЛАВА 3

Сортировка изображений животных

https://t.me/it_boooks/2

Все мы любим картинки. Каждый год мы делаем более триллиона одних только фотографий, а если учитывать рисунки и картины, то и того больше!

Использование компьютеров для сортировки изображений и помощи нам в поиске нужных изображений называется *распознаванием изображений (и образов)*. Для создания системы распознавания изображений нам нужно будет собрать достаточно много изображений одного и того же предмета. Затем мы представим эти изображения для обучения модели машинного обучения, которая определит, что общего у этих изображений, и, используя эти знания, научится распознавать новые изображения.

Например, если мы хотим обучить компьютер распознавать фотографии котят, нужно собрать много фотографий, на которых изображены котята. Система машинного обучения будет использовать эти фотографии для изучения форм и окраса, которые чаще всего встречаются на фотографиях пушистиков. В итоге модель сможет распознать, есть ли котенок на фотографии, которую мы ей покажем.

Люди сталкиваются с распознаванием изображений каждый день. Онлайн-сервисы для обмена фотографиями используют распознавание изображений для сортировки изображений, которые мы загружаем. Веб-сайты используют его для описания фотографий, чтобы люди с нарушениями зрения могли понять, что изображено на этих фотографиях. Социальные сети используют его для распознавания лиц наших друзей и членов семьи на публикуемых нами фотографиях. Компании используют его для отслеживания появления своих логотипов или продуктов на фотографиях, публикуемых в Интернете, чтобы знать, как часто они упоминаются в соцсетях. И главное, распознавание изображений используется в медицине, чтобы помочь врачам распознавать заболевания по снимкам и фотографиям пациентов. Врачам необходимо помнить множество различных симптомов и признаков самых разных болезней, а распознавание изображений может

помочь им обнаружить, например, опухоль кожи на фотографии или рак на микроскопическом изображении клетки.

В этой главе вы создадите собственную систему распознавания изображений и обучите модель машинного обучения распознавать и автоматически сортировать фотографии животных. Давайте начнем!

Выполнение проекта

Для начала выберите два вида животных, которые компьютер должен будет научиться распознавать. Я выбрал коров и овец, чтобы выполнить проект на тему фермы (рис. 3.1). Но можно выбрать и других животных. Главное, чтобы у вас не возникало трудностей при поиске этих фотографий.



Рис. 3.1. Сортировка фотографий животных по группам

Обучение модели

Чтобы обучить компьютер распознавать изображения выбранных вами двух видов животных, вам нужно собрать как можно больше изображений этих животных и использовать их для обучения модели машинного обучения.

1. Создайте новый проект машинного обучения. Назовите его «Сортировка животных», а в качестве распознаваемого типа данных выберите «изображения».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 3.2.

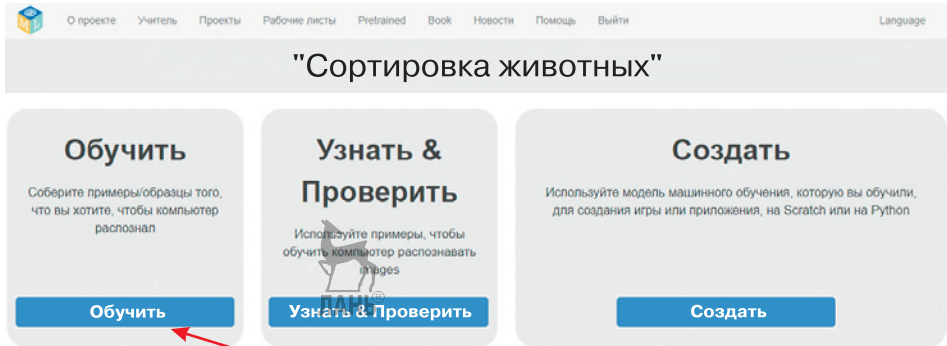


Рис. 3.2. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

3. Нажмите на кнопку «Добавить новую метку» (рис. 3.3) для создания первой коллекции. Затем введите название первого вида животных*.

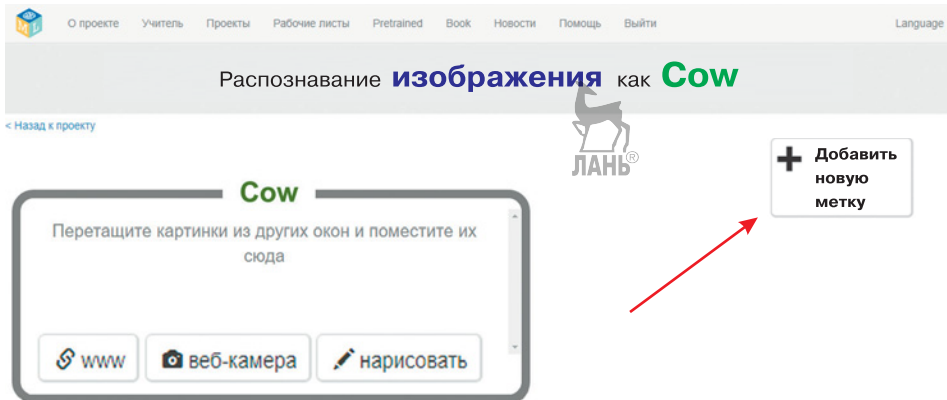


Рис. 3.3. Нажмите на кнопку «Добавить новую метку» для создания новой коллекции примеров

4. Откройте второе окно вашего браузера (обычно в меню есть опция «Новое окно») и расположите эти два окна рядом, как показано на рисунке 3.4. Во втором окне начните поиск фотографий первого вида животных. В моем примере это фотографии коров.

* Сайт не предоставляет возможности создавать имя новой метки на кириллице, поэтому используйте английские слова: cow — корова и sheep — овца (овцы).

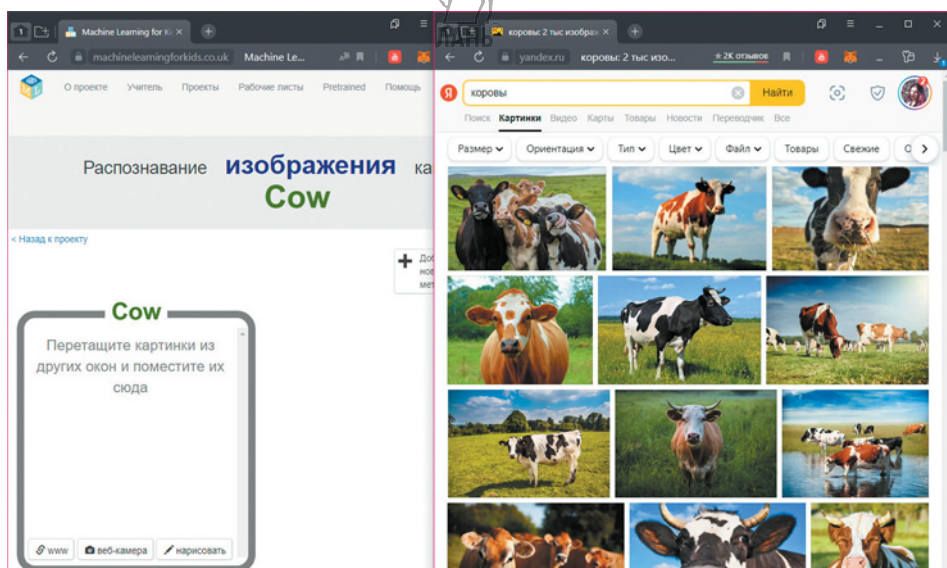


Рис. 3.4. Расположите два окна браузера рядом

5. Перетащите подходящую фотографию из окна поиска прямо в коллекцию в окне проекта. Вы увидите, как в коллекции появилась уменьшенная версия (предпросмотр) этой фотографии (рис. 3.5). Если этого не произошло, попробуйте перетащить фотографию еще раз.

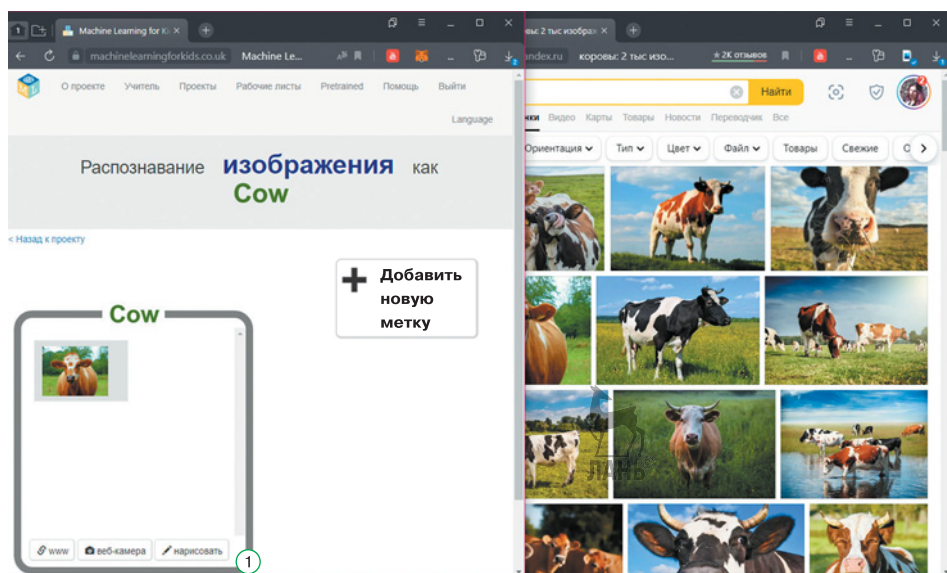


Рис. 3.5. Наполнение коллекции «Cow»

6. Повторяйте шаг 5 до тех пор, пока в коллекции не наберется как минимум 10 разных изображений выбранного вами животного (рис. 3.6).

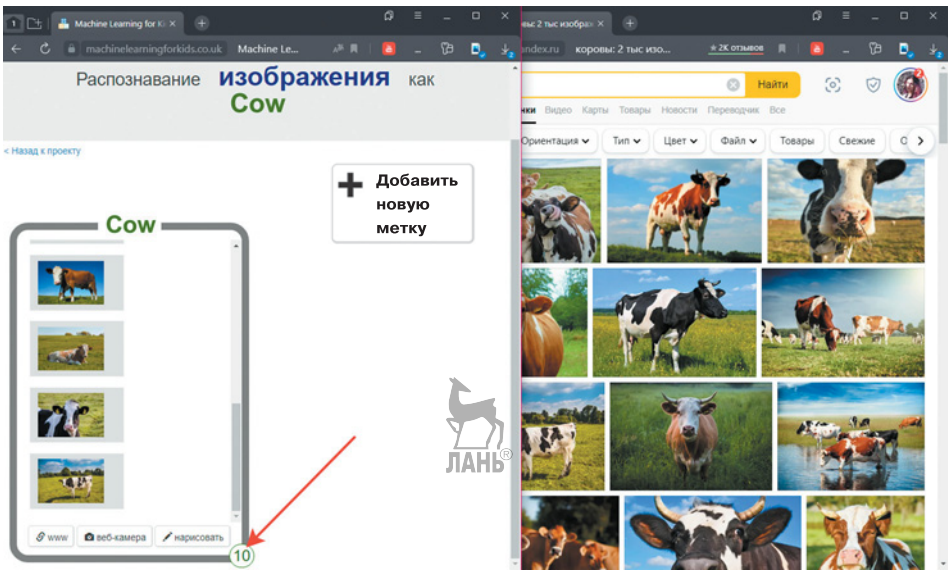


Рис. 3.6. Собранная коллекция из 10 фотографий коров для распознавания

7. Повторяйте шаги 3–6 до тех пор, пока в коллекциях для обоих видов животных не наберется как минимум по 10 изображений, как показано на рисунке 3.7.

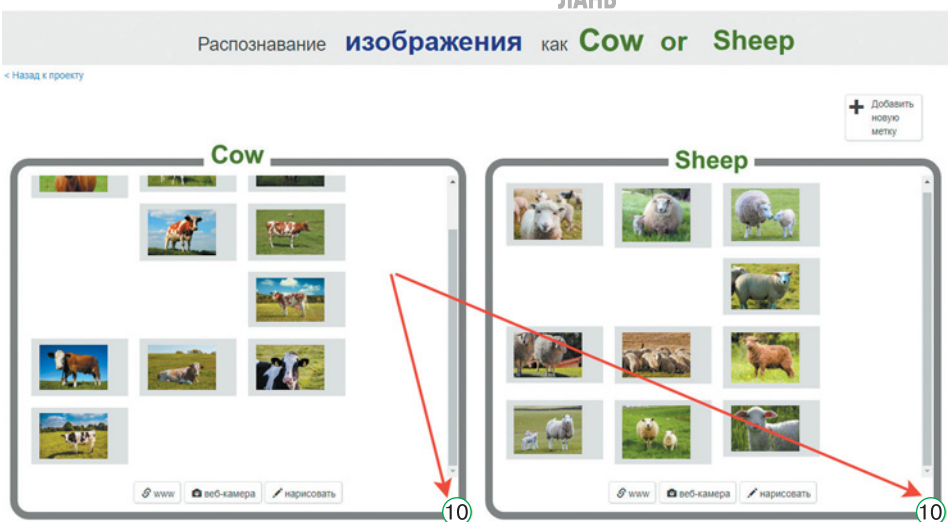


Рис. 3.7. Исходные данные для моего проекта на тему фермы

8. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

9. Нажмите на кнопку «Узнать и проверить» (рис. 3.8), чтобы перейти к следующему этапу проекта.

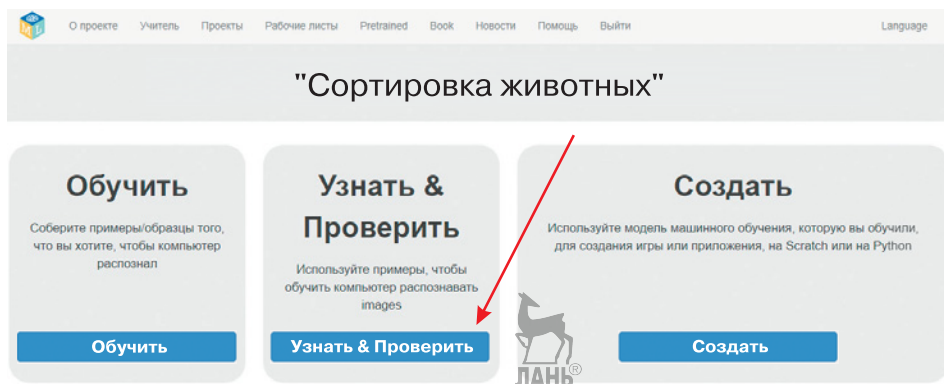


Рис. 3.8. Переход ко второму этапу работы над проектом — этапу «Узнать и проверить»

10. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 3.9).

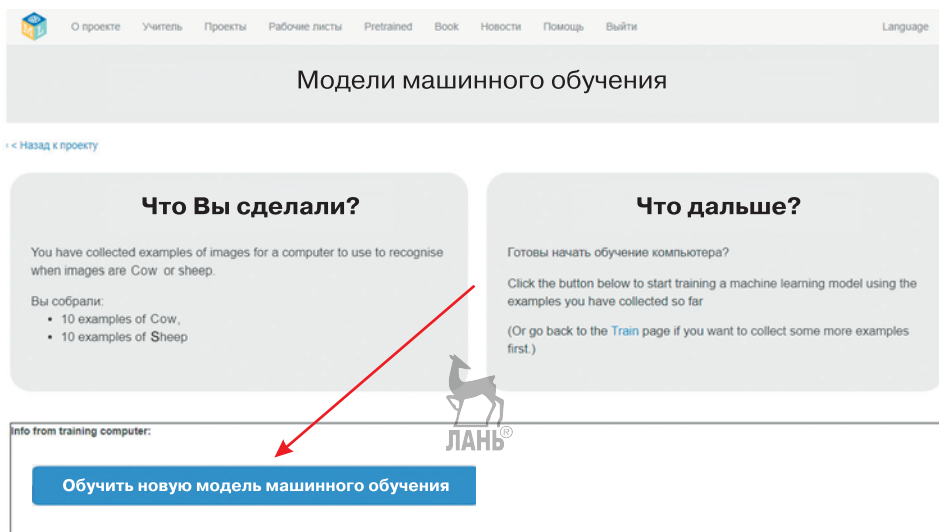


Рис. 3.9. Нажмите на кнопку «Обучить новую модель машинного обучения», чтобы начать обучение

Компьютер будет использовать собранные вами примеры для того, чтобы понять, что общего у фотографий каждого вида животных. Это может занять несколько минут, но, пока ждете, вы можете перейти к следующему шагу и выполнить его во втором окне браузера.

Подготовка проекта

Чтобы протестировать вашу модель машинного обучения, понадобятся еще фотографии, но не из тех, которые вы использовали для обучения модели*. Компьютер будет использовать результаты анализа обучающих примеров, чтобы попытаться распознать одно из двух выбранных вами животных на новых фотографиях. Затем вы создадите проект в Scratch, с помощью которого проверите, насколько хорошо обучилась ваша модель.

1. Найдите еще несколько фотографий выбранных животных и сохраните их на свой компьютер. Чтобы сохранить фотографию, кликните по ней правой кнопкой мыши и выберите «Сохранить изображение» или «Сохранить изображение как...» (рис. 3.10).

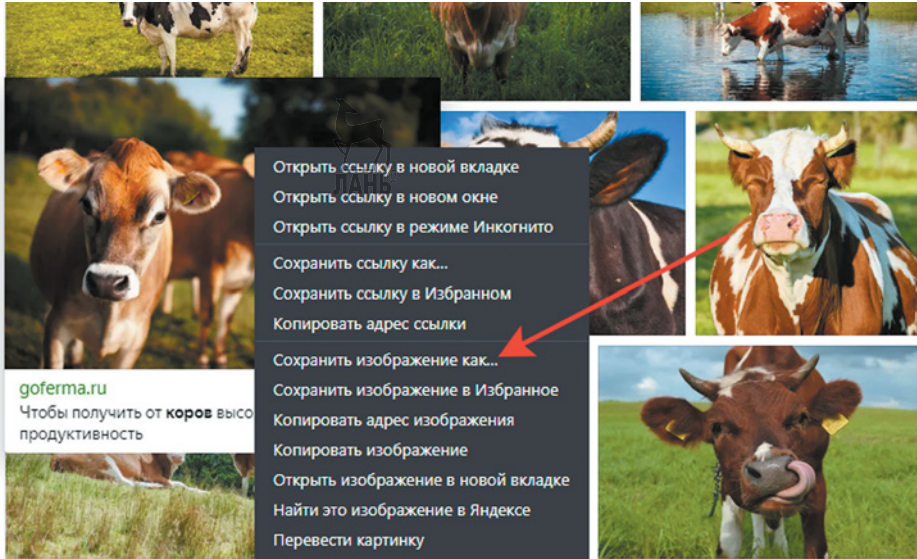


Рис. 3.10. Сохранение изображения на компьютер

* В теории машинного обучения такая выборка будет называться *тестовой*, потому что с ее помощью тестируют качество работы модели. А фотографии, которые мы показали ранее, — это *обучающая* выборка. По ней модель обучалась. — *Прим. ред.*



Примечание

Не используйте те же фотографии, что использовали в качестве примеров для обучения модели. Вам нужно проверить, насколько хорошо компьютер научился распознавать изображения, а не насколько хорошо он научился их запоминать.

2. Сохраните хотя бы по пять фотографий каждого вида животного, как показано на рисунке 3.11.



Рис. 3.11. Моя папка с сохраненными фотографиями коров и овец для тестирования модели

3. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

4. Нажмите на кнопку «Создать» (рис. 3.12).

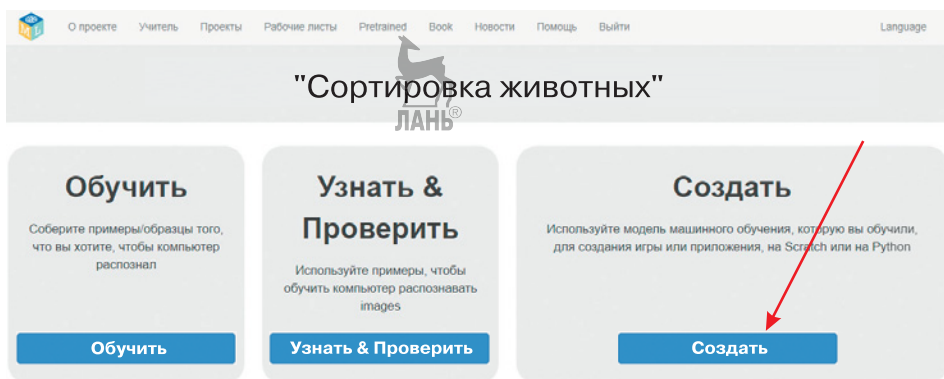


Рис. 3.12. Переход к третьему этапу проекта — этапу «Создать»

5. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch.

На Панели инструментов вы увидите новую категорию блоков, которая будет называться так же, как и ваш проект (рис. 3.13).

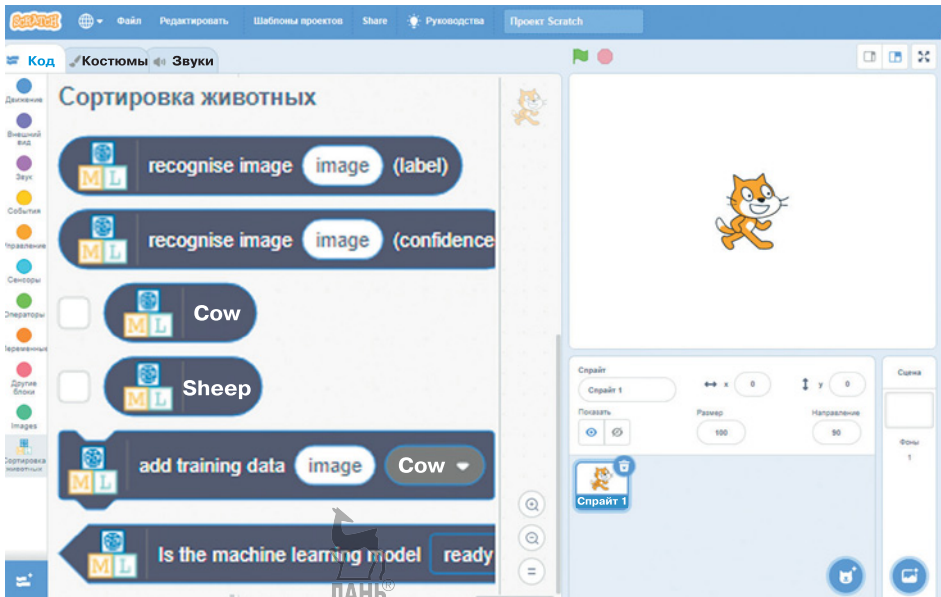


Рис. 3.13. Новые блоки для вашего проекта машинного обучения автоматически добавятся на Панель инструментов Scratch

6. Создайте фон для вашего проекта.

Наведите указатель мыши на иконку «Выбрать фон» в правом нижнем углу окна Scratch (рис. 3.14).

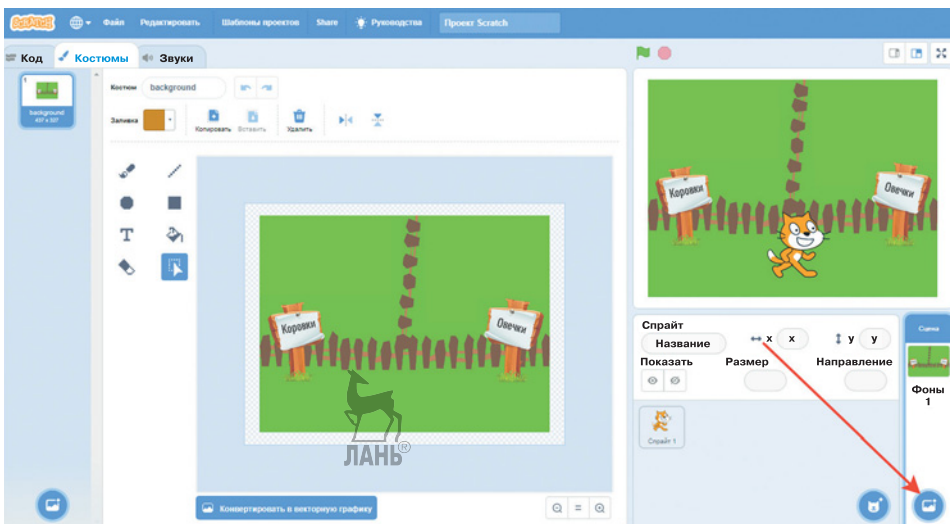


Рис. 3.14. Мой фон для проекта на тему фермы, в котором будут сортироваться коровы и овцы

Далее у вас есть несколько вариантов. Если вам не нравится рисовать самим, можете нажать на кнопку «Выбрать фон» и выбрать один из предлагаемых стандартных фонов или нажать на кнопку «Загрузить фон» и загрузить изображение с вашего компьютера. Также вы можете создать собственный фон для ваших животных. Для этого нажмите на кнопку «Нарисовать» и с помощью инструментов для рисования и раскраски в открывшемся редакторе создайте новый фон для проекта.

Вы можете выбрать любой из вариантов, но главное убедитесь, что на вашем фоне есть четко обозначенные участки для каждого вида животных.

Я выбрал животных, которые обычно живут на ферме, поэтому в качестве фона нарисовал ферму и загоны с надписями «Коровки» и «Овечки». Вы можете нарисовать что-то другое, что будет соответствовать животным, которых выбрали вы. Например, если это собаки и кошки, фоном может быть зоомагазин, а если это львы и слоны — зоопарк.

7. Нажмите на спрайт кота, а затем наведите указатель мыши на иконку «Выбрать костюм» в левом нижнем углу экрана. Нажмите на кнопку «Загрузить костюм», как показано на рисунке 3.15.

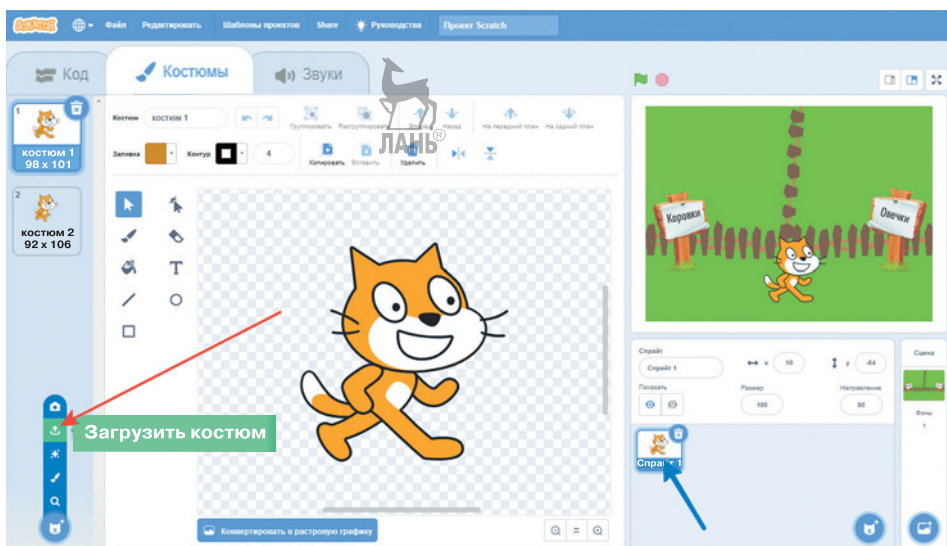


Рис. 3.15. Нажмите на кнопку «Загрузить костюм», чтобы добавить тестовые изображения. Вы также можете удалить костюм кота из панели костюмов в левой части экрана

8. Выберите все тестовые изображения, которые вы скачали на шаге 2, чтобы загрузить их одновременно.



Примечание

Убедитесь, что вы делаете это именно для спрайта, а не для фона.

9. Если вы случайно выбрали не все тестовые изображения при загрузке, нажмите на кнопку «Загрузить костюм» снова и повторяйте это до тех пор, пока в проект не будут загружены все изображения из шага 2.

Вам не понадобится стандартный костюм кота, поэтому вы можете удалить его. Для этого щелкните по нему мышкой на панели костюмов, которая находится в левой части экрана (см.рис. 3.15), а затем нажмите на иконку корзины в правом верхнем углу.

Также убедитесь, что вы загружаете все тестовые изображения в качестве костюмов к одному спрайту, как показано на рисунке 3.16, а не каждое изображение к отдельному спрайту.

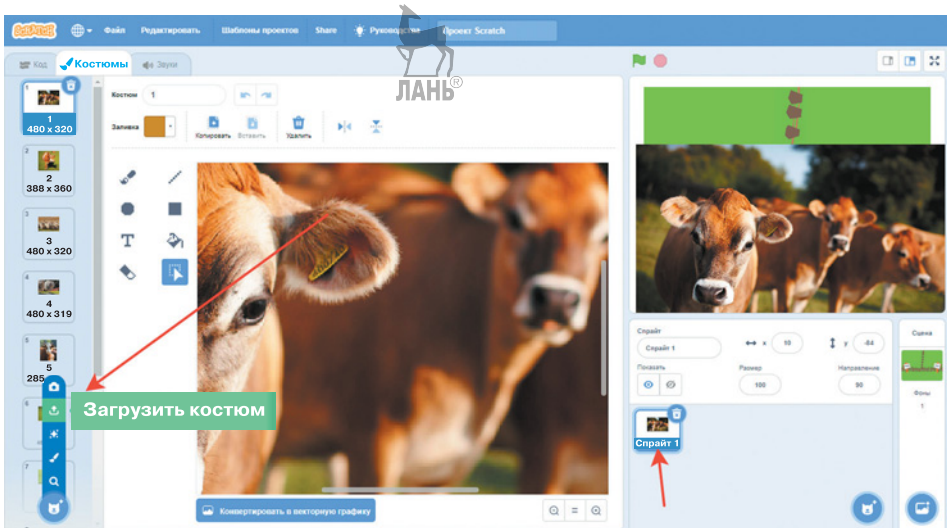


Рис. 3.16. Загрузка дополнительных костюмов к одному и тому же спрайту

10. Перейдите на вкладку «Код» и создайте такой же скрипт, как показан на рисунке 3.17.



Примечание

Если вы не знаете, как создаются скрипты в Scratch, возвращитесь к разделу «Программирование в среде Scratch» на с. 15.

Этот скрипт будет рассматривать каждый из костюмов с вашими тестовыми изображениями и использовать созданную вами

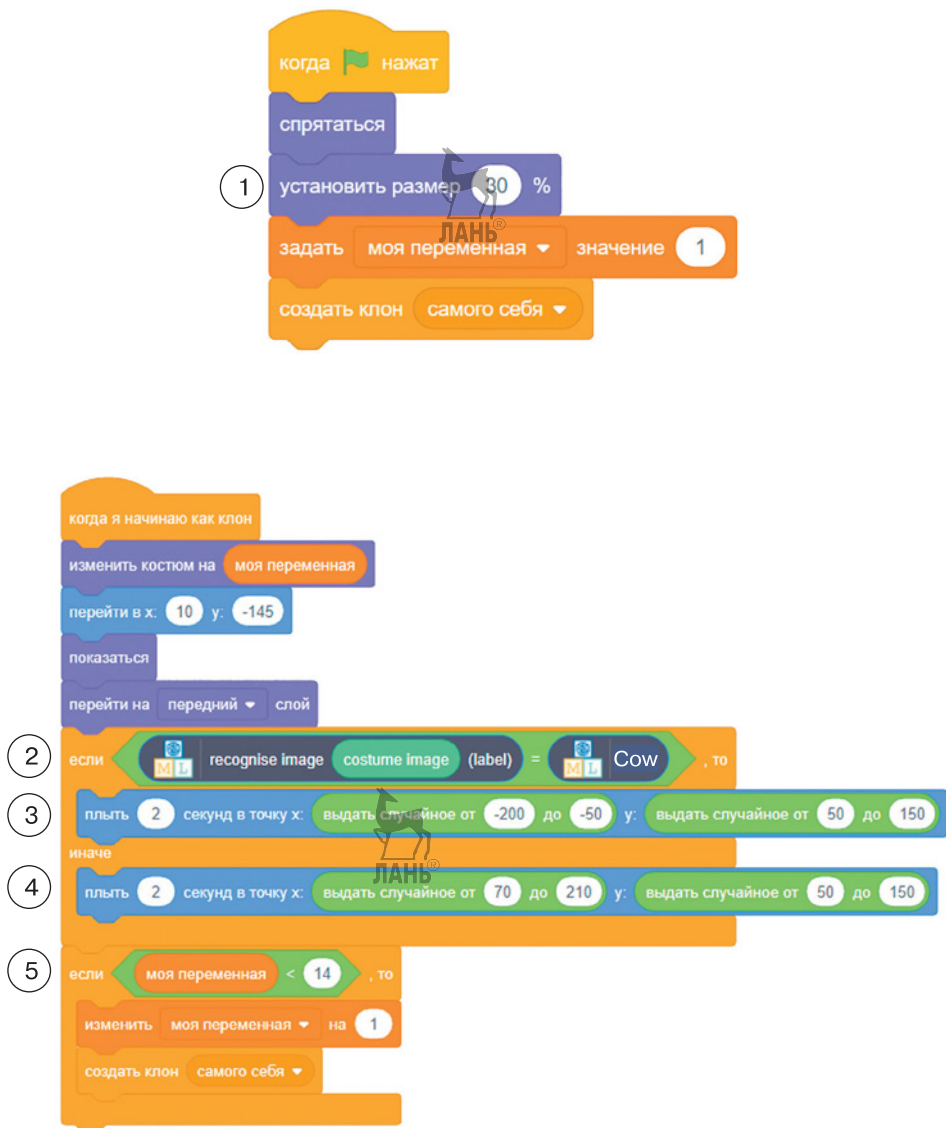


Рис. 3.17. Пример скрипта для сортировки фотографий животных

модель машинного обучения, чтобы распознать животное и переместить его в нужный раздел.

Блок «перейти в x: 10 y: -145» ① задает начальное расположение каждого изображения на экране. Согласно этому скрипту каждая новая фотография животного будет появляться по центру

нижней части экрана. Вы можете изменить эти координаты, чтобы начальное положение изображений соответствовало рисунку выбранного вами фона.

Блок «recognise image»* ② использует вашу модель машинного обучения для распознавания изображения.

Блок «плыть 2 секунды в точку x: выдать случайное от -200 до -50 y: выдать случайное от 50 до 150» ③ перемещает фотографию и задает ей случайное расположение в левом верхнем углу экрана. Вы можете изменить эти координаты и задать координаты той области экрана, в которой вы хотите расположить отсортированные изображения животных первого вида.

Блок «плыть 2 секунды в точку x: выдать случайное от 70 до 210 y: выдать случайное от 50 до 150» ④ перемещает фотографию и задает ей случайное расположение в правом верхнем углу экрана. Вы можете изменить эти координаты и задать координаты той области экрана, в которой вы хотите расположить отсортированные изображения животных второго вида.

Число в блоке «если моя переменная < 14 » ⑤ зависит от того, сколько тестовых изображений вы загрузили в проект. Задайте в этом блоке значение, которое соответствует количеству костюмов, загруженных вами на шаге 8. Я загрузил 14 костюмов для тестового спрайта, поэтому мой скрипт проанализирует 14 фотографий.

Тестирование модели

Чтобы протестировать вашу модель в только что созданном проекте, нажмите на зеленый флажок в левом верхнем углу, как показано на рисунке 3.18. Ваша модель машинного обучения отсортирует загруженные вами тестовые изображения животных на две группы.

Посчитайте, сколько изображений модель переместила в правильную сторону. Это самый простой способ определить, насколько хорошо ваш проект справляется с сортировкой изображений двух видов животных.

Если ваша модель ошибается, вы можете попытаться ее улучшить, увеличив число обучающих примеров. Вернитесь к этапу «Обучить» и добавьте в каждую коллекцию больше фотографий. Затем перейдите к этапу «Узнать и проверить» и обучите новую, улучшенную модель. После этого снова запустите скрипт Scratch на выполнение и проверьте, стала ли обновленная модель сортировать фотографии лучше.

* В переводе с англ.: распознать изображение. — Прим. пер.

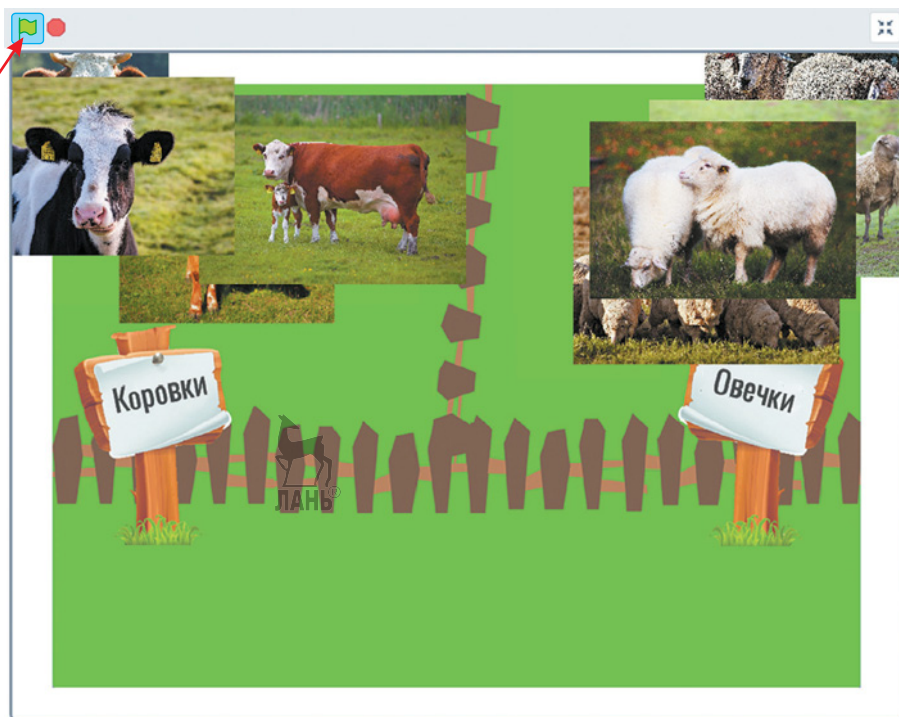


Рис. 3.18. Распознавание изображений и сортировка их на две группы

Обзор и улучшение проекта

Вы только что успешно обучили модель машинного обучения распознавать изображения животных! И в основе этого проекта не лежат четкие правила и пошаговые инструкции. Вы не составляли подробных описаний, как выглядят разные животные, не давали компьютеру конкретных инструкций, как их распознавать. Вместо этого вы использовали машинное обучение. Такой подход известен как **контролируемое обучение** (supervised learning), потому что вы контролировали процесс, когда собирали коллекции обучающих примеров.

И до тех пор, пока ваши тестовые изображения будут похожи на те, что использовались для обучения, модель должна работать. Однако, если вы попытаетесь протестировать ее с использованием изображений, которые существенно отличаются от обучающих примеров, скорее всего вы получите совсем другие результаты.

Например, я попытался заменить костюмы в своем проекте на рисунки коров и овец вместо их фотографий. Затем я снова запустил проект, нажав на зеленый флажок. Как вы можете видеть на рисунке 3.19, моя модель допустила много ошибок.

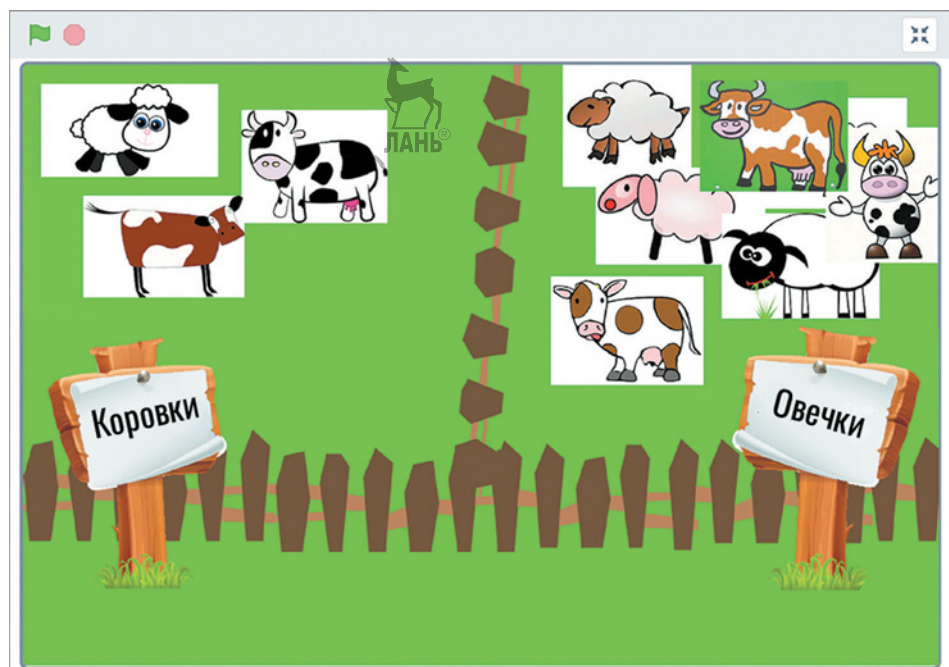


Рис. 3.19. Модели машинного обучения ошибаются, если тестовые примеры не похожи на обучающие примеры

Я получил такие результаты, потому что те черты и шаблоны, которые моя модель научилась распознавать на фотографиях, не помогли ей распознавать очертания рисунков. Если вы хотите, чтобы компьютер мог распознавать и фотографии, и рисунки, вам нужно добавить в обучающие примеры и то и другое.

Вернитесь к этапу «Обучить» и обновите коллекции обучающих примеров, чтобы в них были и фотографии, и рисунки, как показано на рисунке 3.20. Я собрал по 10 примеров фотографий и рисунков в каждой коллекции.

Затем перейдите к этапу «Узнать и проверить» и обучите обновленную модель, использующую новый набор обучающих примеров. Этот новый набор должен помочь компьютеру научиться определять общие черты на фотографиях и рисунках, а затем распознавать и то и другое. На рисунке 3.21 показано, насколько лучше работает моя обновленная модель.

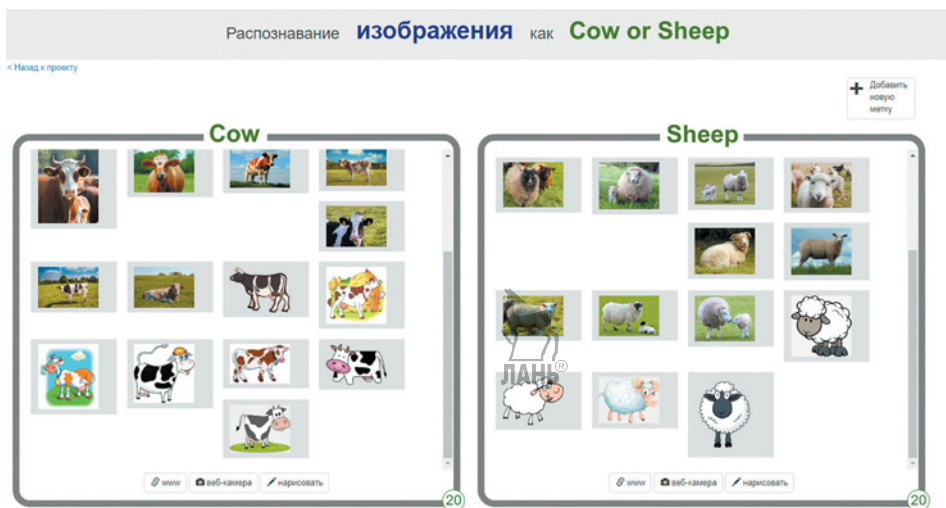


Рис. 3.20. Обучение компьютера распознавать фотографии и рисунки одновременно

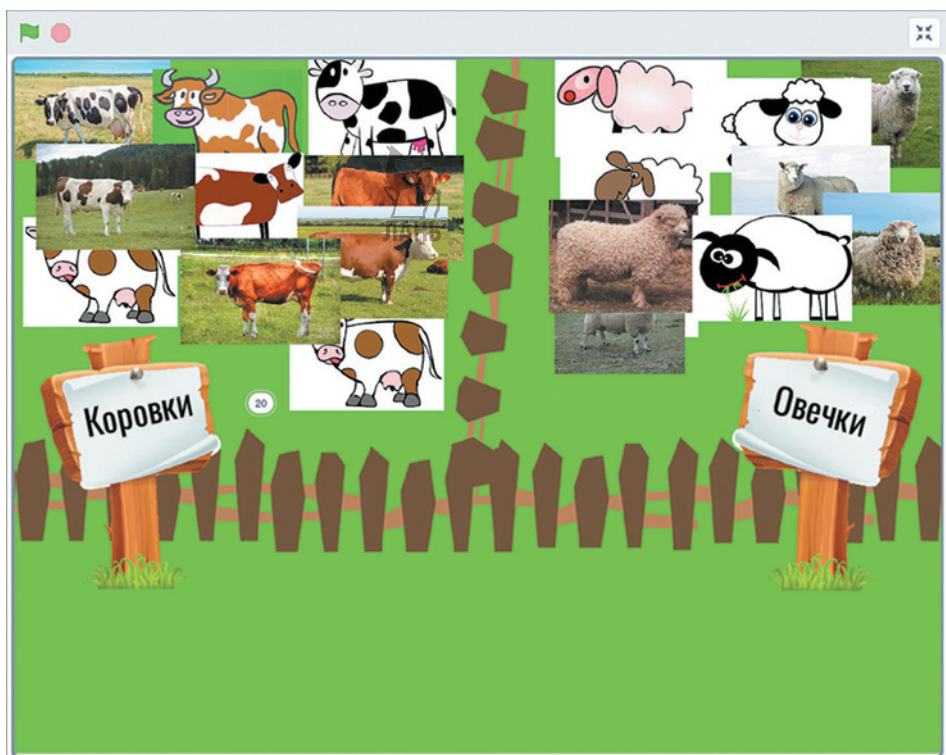


Рис. 3.21. Тестирование модели с использованием и фотографий, и рисунков

Как видите, чем больше тестовые изображения похожи на те, что использовались в качестве обучающих примеров, тем лучше работает модель машинного обучения.

А какие еще изменения можно внести, чтобы улучшить вашу модель?

Что вы узнали

В этой главе вы использовали машинное обучение для создания системы распознавания изображений, которая умеет распознавать и сортировать изображения животных. Вы узнали о некоторых ключевых принципах работы над проектами машинного обучения, например о том, что результаты будут лучше, если увеличить количество обучающих примеров и использовать для обучения изображения, похожие на те, которые компьютер должен уметь распознавать.

Вы также узнали, что можно оценить, насколько хорошо работает система распознавания изображений, если опробовать ее на тестовых примерах и посчитать, сколько из них она распознала правильно. Вы убедились в этом, создав проект в Scratch и проверив, насколько хорошо ваша модель машинного обучения сортирует фотографии животных. Вы познакомились с понятием контролируемого машинного обучения.

В следующей главе вы обучите другую систему распознавания изображений и используете ее для создания игры. Вы также узнаете о некоторых случаях, когда в проектах машинного обучения что-то может пойти не по плану. Бывает и так!





ГЛАВА 4

Игра с компьютером в «Камень, ножницы, бумага»

В главе 3 вы использовали машинное обучение, чтобы создать систему распознавания, которая умеет сортировать изображения животных. Вы узнали, что для создания такой системы нужно собрать достаточно большое количество примеров изображений, которые компьютер должен научиться распознавать.

В этой главе вы обучите модель машинного обучения распознавать жесты, которые показывают игроки в «Камень, ножницы, бумага» (рис. 4.1), а затем запрограммируете компьютер, чтобы он мог играть в эту игру против вас.

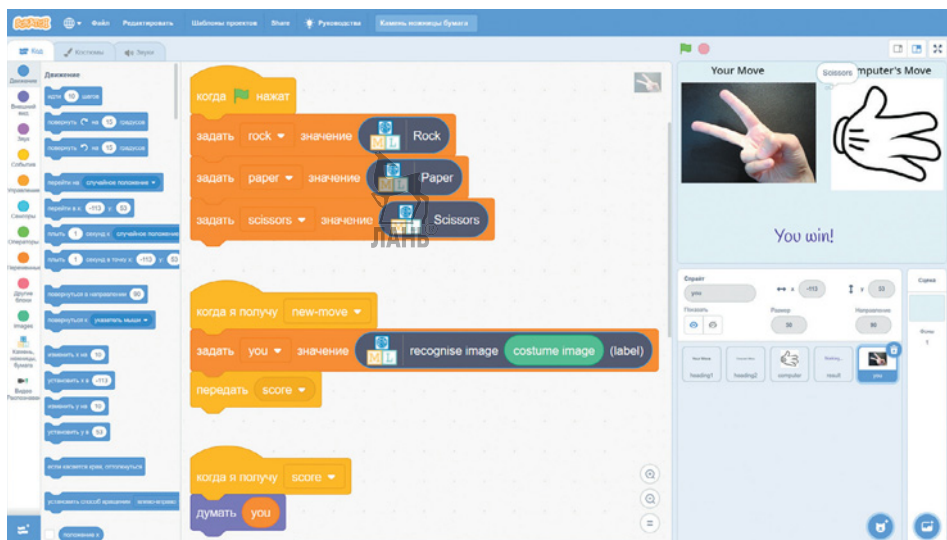


Рис. 4.1. Игра в «Камень, ножницы, бумага» с компьютером

Давайте начнем!

Выполнение проекта

При выполнении этого проекта вы будете делать фото своих рук, поэтому вам понадобится веб-камера.



Примечание

В качестве веб-камеры к компьютеру можно подключить мобильный телефон. Попросите родителей или учителей помочь вам с этим.

Обучение модели

1. Создайте новый проект машинного обучения и назовите его «Камень, ножницы, бумага». В качестве распознаваемого типа данных выберите «изображения».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 4.2.

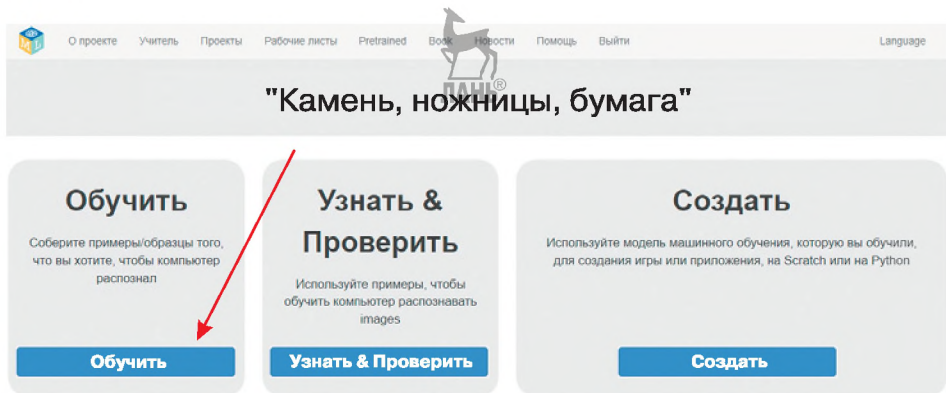


Рис. 4.2. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

3. Нажмите на кнопку «Добавить новую метку» (рис. 4.3) для создания первой коллекции, назовите ее «Rock» (англ.: камень). Затем создайте еще две коллекции — «Paper» (англ.: бумага) и «Scissors» (англ.: ножницы), как показано на рисунке 4.3.

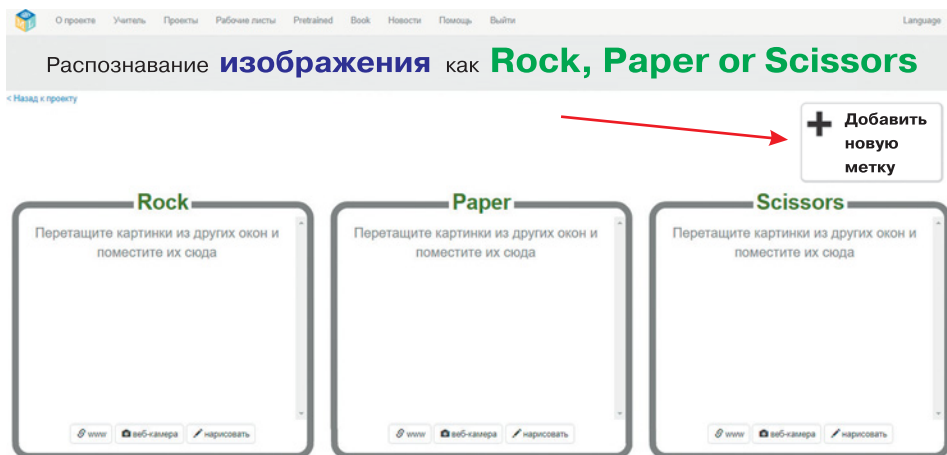


Рис. 4.3. Нажмите на кнопку «Добавить новую метку» для создания новой коллекции примеров

4. Нажмите на кнопку «веб-камера» в коллекции «Rock» и сфотографируйте свою руку, сжатую в кулак, как показано на рисунке 4.4.



Примечание

Когда вы будете делать это в первый раз, ваш браузер скорее всего запросит разрешение на использование веб-камеры. Во всплывающем окне нажмите кнопку «Разрешить».

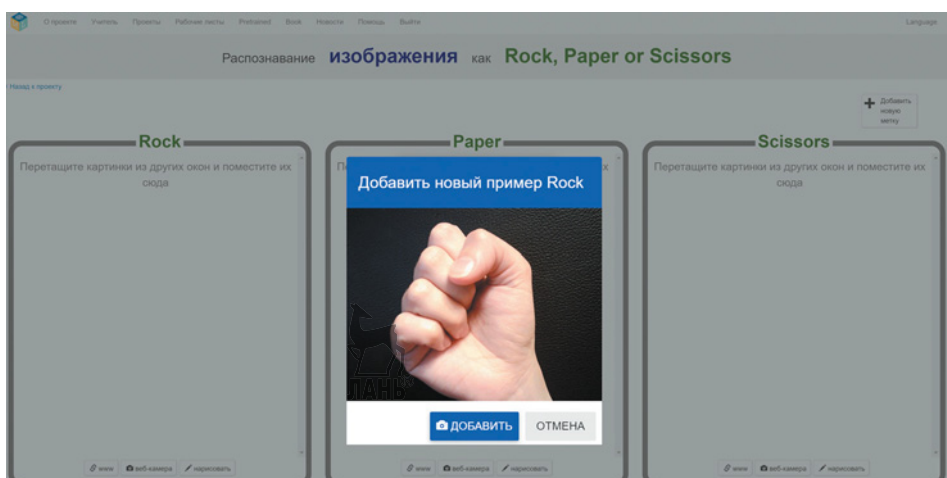


Рис. 4.4. Тестовые примеры для проекта, полученные с помощью веб-камеры

5. Когда будете готовы, нажмите на кнопку «Добавить» (рис. 4.4). В коллекции «Rock» должно появиться первое фото вашего кулака (рис. 4.5).

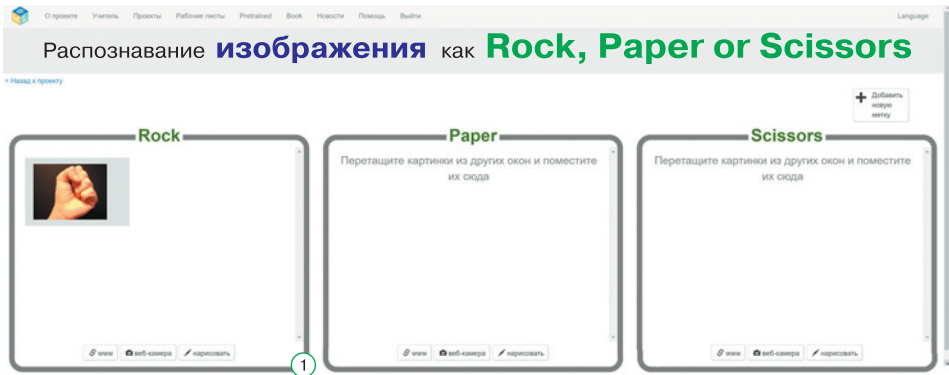


Рис. 4.5. Миниатюра вашей фотографии появится в коллекции

6. Повторяйте шаги 4 и 5 для наполнения коллекций «Paper» и «Scissors». Делайте это до тех пор, пока в коллекции «Rock» не наберется хотя бы 10 фото кулаков, обозначающих камень, в коллекции «Paper» — хотя бы 10 фото разжатых ладоней, обозначающих бумагу, а в коллекции «Scissors» — хотя бы 10 фото жеста, обозначающего ножницы (два пальца подняты вверх), как показано на рисунке 4.6.

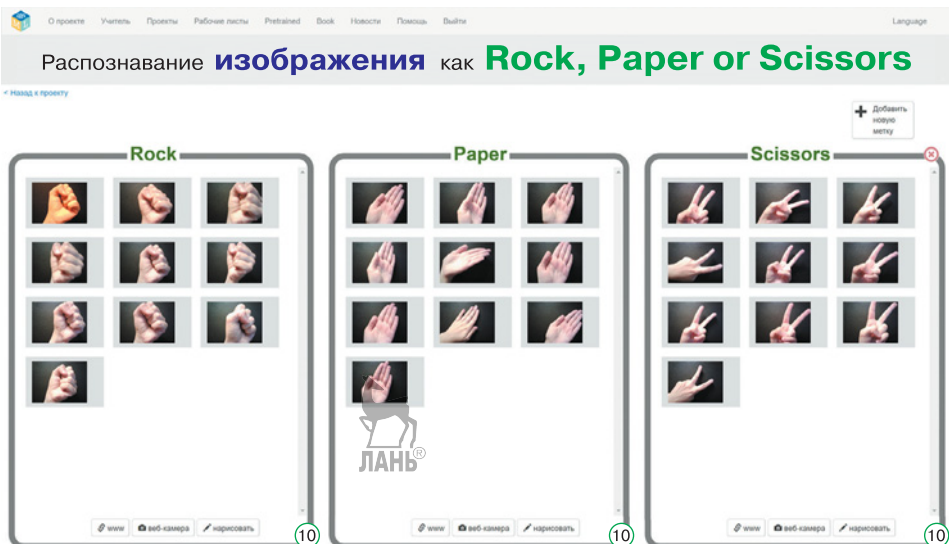


Рис. 4.6. Обучающие примеры для коллекций «Rock», «Paper» и «Scissors»

7. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

8. Нажмите на кнопку «Узнать и проверить» (рис. 4.7), чтобы перейти к следующему этапу проекта.

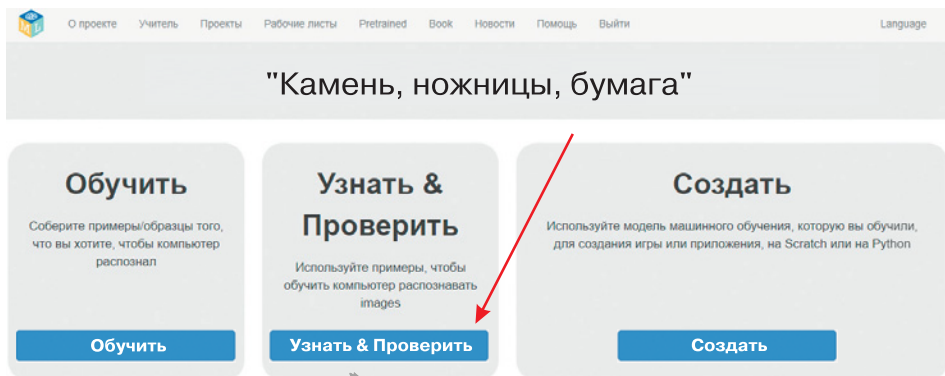


Рис. 4.7. Переход ко второму этапу работы над проектом — этапу «Узнать и проверить»

9. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 4.8).

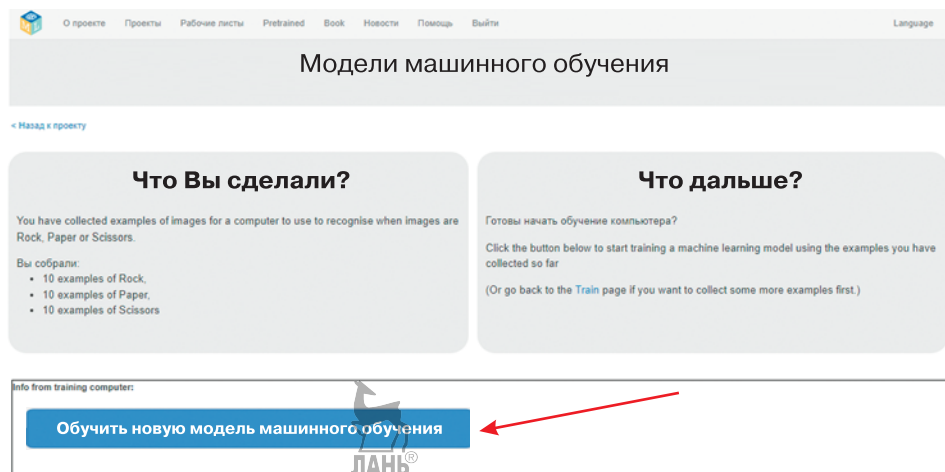


Рис. 4.8. Нажмите на кнопку «Обучить новую модель машинного обучения», чтобы начать обучение

Сделанные вами фотографии будут использоваться для обучения модели машинного обучения. Компьютер изучит, что общего у фотографий в каждой коллекции, и таким образом научится распознавать жесты. Здорово, правда? Этот процесс может занять несколько минут, но, пока вы ждете завершения обучения моде-

ли, вы можете перейти к следующему шагу и начать создавать собственную игру.

Подготовка проекта-игры

Вы создадите скрипт на языке Scratch, который будет использовать вашу модель машинного обучения, чтобы играть с вами в «Камень, ножницы, бумага». Этот скрипт с помощью веб-камеры будет делать фотографии вашей руки, а модель машинного обучения — распознавать жест, который вы показали.

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

2. Нажмите на кнопку «Создать».

3. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch.

На Панели инструментов вы увидите новую категорию блоков, которая будет называться так же, как и ваш проект.

4. В Главном меню Scratch (в верхней части экрана) выберите **Шаблоны проектов** (рис. 4.9).

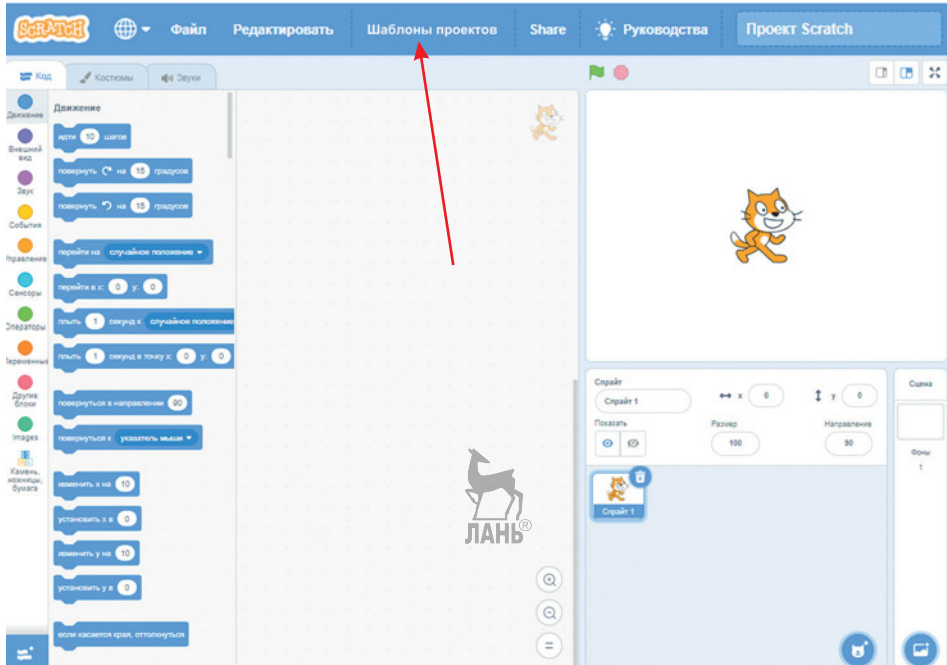


Рис. 4.9. Новые блоки для вашего проекта машинного обучения автоматически добавятся на Панель инструментов Scratch

Вам будет предоставлена возможность доступа к различным примерам и шаблонам проектов для экономии вашего времени.

5. Выберите шаблон «Rock, Paper, Scissors» в списке шаблонов проектов. Чтобы быстрее его найти, вы можете начать вводить это название в строке поиска или выбрать категорию «Images projects» в верхней части страницы.

Этот шаблон содержит частично созданную игру «Камень, ножницы, бумага» на языке Scratch. Выполните следующие шаги, чтобы добавить в проект вашу модель машинного обучения. Но прежде чем начать, внимательно прочитайте скрипт и попытайтесь понять, как он работает.

6. Нажмите на спрайт «you» (англ.: вы/ты) (рис. 4.10) и найдите фрагменты скрипта «когда зеленый флажок нажат» (показан на рисунке синей стрелкой) и «когда я получу новый ход» (показан красной стрелкой).

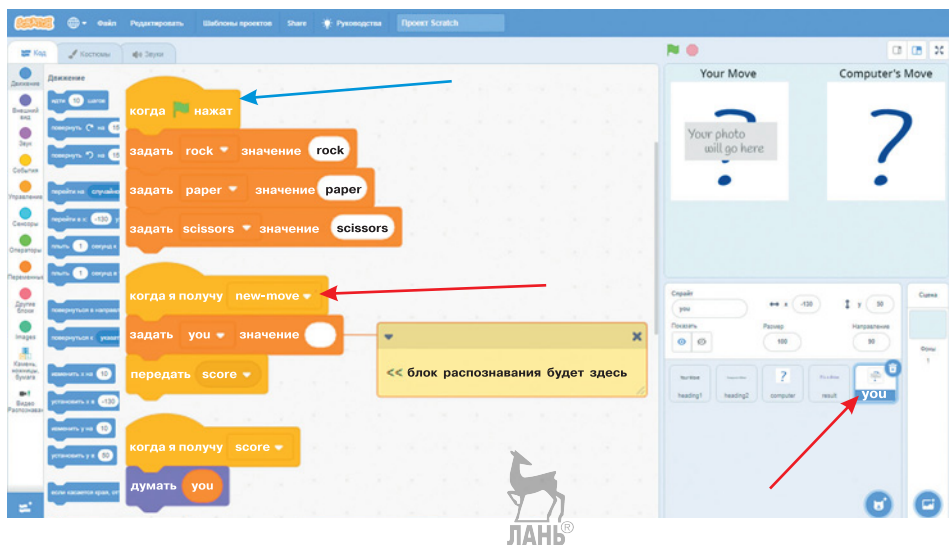


Рис. 4.10. Найдите фрагменты скрипта, которые нужно изменить для спрайта «you»

7. Во фрагмент скрипта «когда зеленый флажок нажат» (синяя стрелка на рисунке 4.11) переместите блоки с названиями коллекций «Rock», «Paper» и «Scissors» из вашего проекта машинного обучения, как показано на рисунке 4.11 красными стрелками.

8. Во фрагмент скрипта «когда я получу новый ход» (указан синей стрелкой на рисунке 4.12) переместите блок «recognize»

image» (в переводе с англ.: распознать изображение) и туда же добавьте еще один блок — «costume image» (англ.: изображение костюма) из категории «Images» (англ.: изображения), как показано красными стрелками на рисунке 4.12.

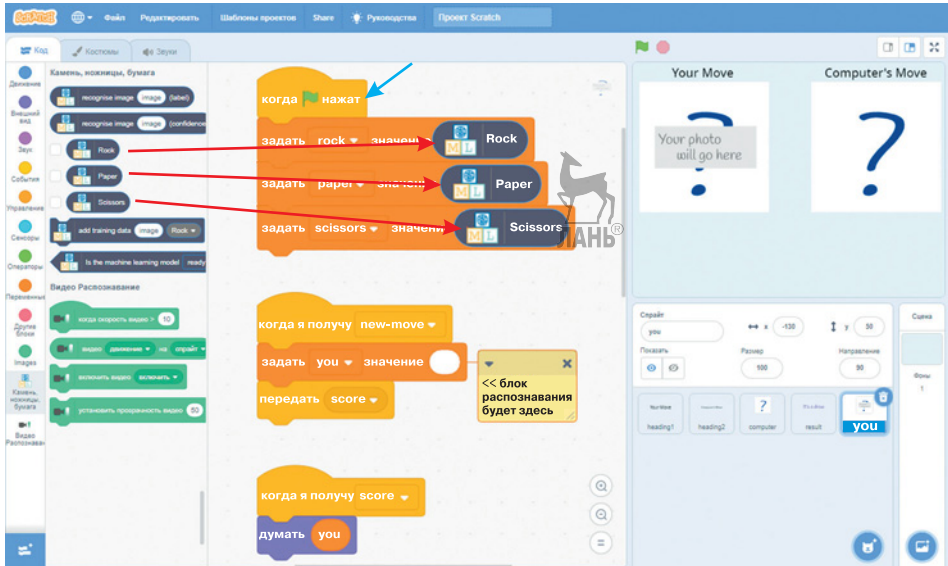


Рис. 4.11. Обновите фрагмент скрипта «когда зеленый флажок нажат», добавив в него блоки из вашего проекта машинного обучения



Рис. 4.12. Обновите фрагмент скрипта «когда я получу новый ход», добавив в него блоки из вашего проекта машинного обучения

Настало время играть! То есть запустить и протестировать ваш проект.

Поднесите руку к веб-камере и покажите жест для камня, для ножниц или для бумаги, а затем нажмите клавишу **P** на клавиатуре (в английской раскладке), чтобы сделать фотографию.

Теперь очередь компьютера делать свой ход. Он случайным образом выберет один из вариантов (камень, ножницы или бумагу), и на экране отобразится соответствующая картинка. Затем он использует модель машинного обучения, чтобы распознать ваш жест, определит, кто выиграл, и отобразит сообщение об этом (рис. 4.13).

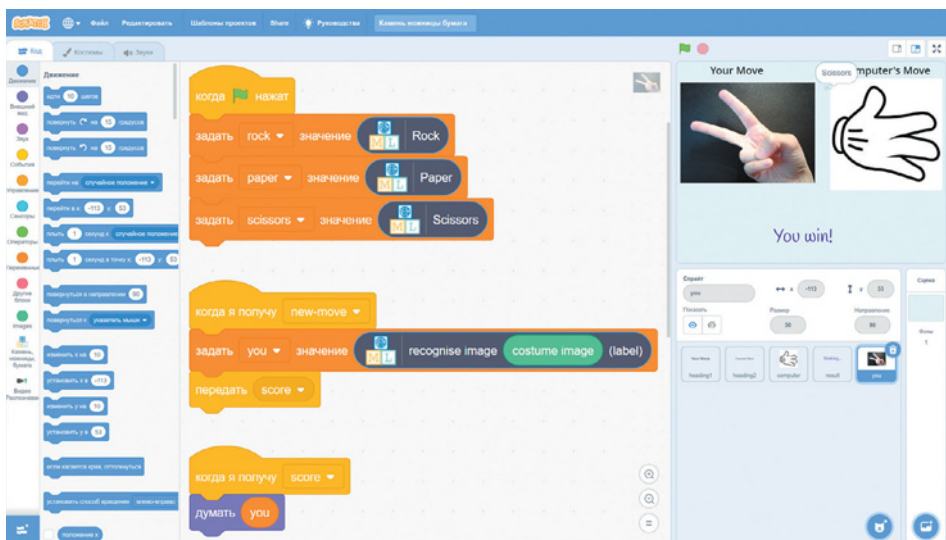


Рис. 4.13. Игра в «Камень, ножницы, бумага» с компьютером

Обзор и улучшение проекта

Итак, вы обучили модель машинного обучения распознавать три разных вида жестов. Здорово! Теперь попробуйте поэкспериментировать с проектом и посмотреть, до каких пор он будет работать правильно, а в какой момент начнет выдавать ошибки.

Помните, что ваша модель машинного обучения на самом деле не научилась играть в «Камень, ножницы, бумага», не стала по-

нимать смысл этой игры и тем более она не начала понимать смысл жестов. Она научилась лишь распознавать очертания на изображениях, которые вы собрали в качестве примеров.

Представьте себе, что вы сделали все фотографии кулаков для коллекции «Rock», держа руку очень близко к веб-камере, так, что она выглядит очень большой. А на всех фотографиях для коллекции «Scissors» ваша рука, наоборот, выглядит очень маленькой, потому что вы держали ее далеко от камеры. В такой ситуации компьютер может подумать, что главным отличием является размер руки на фотографии, а не жест, который эта рука показывает. Тогда при тестировании модели любая фотография, на которой рука изображена крупным планом, будет распознаваться как «камень», независимо от того, какой жест показан на самом деле.

Давайте представим еще одну ситуацию — на всех фотографиях для коллекции «Rock» ваша рука направлена слева направо, а на всех фотографиях для коллекции «Paper» — справа налево, как показано на рисунке 4.14. В этом случае компьютер может предположить, что главным отличием является направление руки, и все фотографии, на которых рука направлена влево, будут распознаваться как «бумага», а фотографии, на которых рука направлена вправо, — как «камень», независимо от показанного жеста.

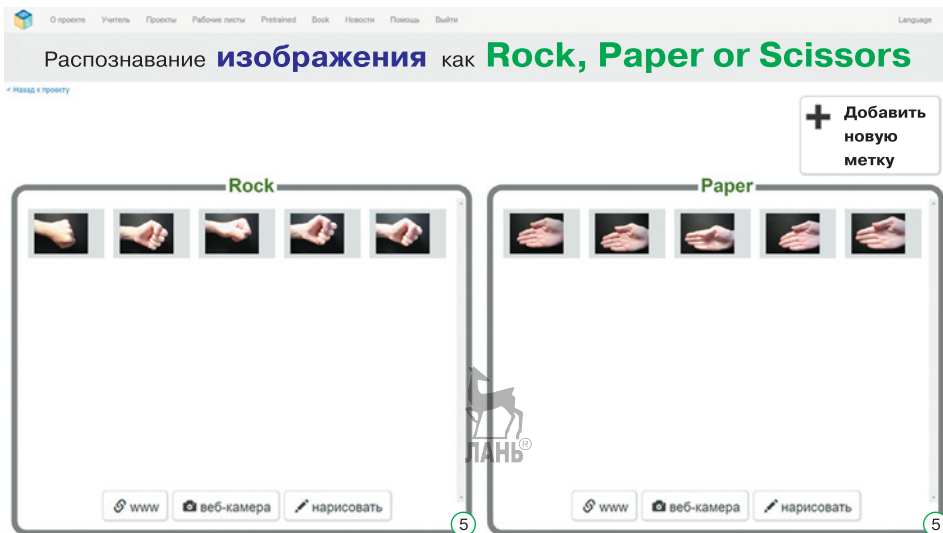


Рис. 4.14. На самом деле компьютер во время обучения будет ориентировать не на те очертания, о которых мы думали изначально

Названия коллекций при этом не имеют никакого значения. Мы задаем эти названия только для того, чтобы упростить организацию проектов для себя, но компьютер не обращает на них внимания, когда ищет общие закономерности обучающих примеров. Если же вы закроете чем-нибудь названия коллекций на рисунке 4.14, покажете их своему другу и попросите своего друга угадать, что общего в этих наборах фотографий, возможно, он вам ответит: «В одном наборе все руки направлены влево, а в другом — вправо». Модель машинного обучения может сработать точно так же. Она тоже не знает смысл, который вы пытаетесь вложить в действия.

Такое же значение может иметь и фон фотографий. Однажды мы с моим учеником обнаружили такую ошибку в его проекте: на всех фотографиях для коллекций «Rock» и «Paper» на заднем плане было его лицо, а на всех фотографиях для коллекции «Scissors» — его и его одноклассника.

Когда он тестировал свой проект в Scratch, все шло отлично и компьютер успешно распознавал все жесты до тех пор, пока я не подошел и не встал рядом с ним, чтобы посмотреть, как все работает. С этого момента, какой бы жест он ни показывал, компьютер распознавал его как «ножницы».

Он даже не догадывался, что на самом деле обучил свою модель машинного обучения распознавать не жесты, а количество людей на фотографии. В итоге все фотографии, на которых было два лица, распознавались как «ножницы».

Если же вы не хотите, чтобы модель машинного обучения взяла за основу не те закономерности, которые вы задумали, постарайтесь разнообразить обучающие примеры в пределах одной коллекции. Ваш проект будет работать гораздо лучше, если обучающие примеры будут включать в себя самые разные фотографии одного и того же предмета. Например, наполняя коллекцию «Rock», фотографируйте свой кулак со всех возможных ракурсов. Сделайте несколько фотографий, когда он приближен к веб-камере и смотрится большим, и еще несколько — где он дальше от веб-камеры и кажется маленьким. Если есть возможность сделать фотографии с разным фоном — это просто отлично. Компьютер научится распознавать именно жест только в том случае, если этот жест будет единственной общей чертой всех фотографий.

В главе 14 вы узнаете больше о том, как «запутать» модель машинного обучения, а пока запомните: *если все изображения в коллекции имеют только одну общую черту, то модель научится распознавать именно ее.*

Что вы узнали

Выполнив проект этой главы, вы обучили еще одну модель машинного обучения распознавать изображения. В главе 3 вы использовали эту возможность для сортировки фотографий, а здесь вы разработали игру «Камень, ножницы, бумага», в которую можно будет играть с компьютером, поскольку он научился распознавать ваши жесты. Оба проекта связаны с распознаванием изображений и являются хорошими примерами повседневного использования этой технологии.

Вы также узнали, что основной способ обучения компьютера распознаванию изображений заключается в сборе большого количества фотографий-примеров. А еще вы усвоили важный урок: чтобы результаты работы проекта были лучше, нужно подбирать обучающие примеры так, чтобы их объединяла только одна общая особенность.

Однако компьютеры могут научиться распознавать не только очертания на фотографии, и в следующей главе вы узнаете о других характерных признаках, которые могут выявлять модели машинного обучения.





ГЛАВА 5

Распознавание постеров фильмов

В предыдущих двух главах вы собирали обучающие примеры (изображения) и создавали модели машинного обучения, способные распознать изображения определенного объекта по цветам, очертаниям или другим общим характерным признакам.

В этой главе вы будете использовать ту же технологию, но модель должна будет обучиться распознавать не конкретный объект на изображении, а стиль этого изображения. Например, если вы наполните одну коллекцию примерами акварельных рисунков, а другую примерами рисунков карандашом, вы сможете создать модель машинного обучения, которая будет распознавать, является ли тестовое изображение акварельным рисунком или карандашным наброском.

Самый распространенный пример того, как эта технология используется в реальной жизни, — поисковые системы. Системы поиска изображений умеют распознавать их стиль, что позволяет фильтровать результаты поиска по типу изображения (картинка, рисунок, фотография и так далее). Такие системы используют модели машинного обучения, обученные на множестве примеров изображений разных стилей.

Некоторые люди используют системы машинного обучения для создания совершенно новых изображений. Например, обучают компьютер распознавать общие закономерности в произведениях искусства определенного стиля, а затем создавать новые произведения в этом стиле. В 2018 году система искусственного интеллекта создала картину, которая затем была продана на аукционе как произведение искусства более чем за 400 000 долларов США.

Такие проекты называются *вычислительным творчеством*, и они используются для создания самых разных вещей. Например, системы искусственного интеллекта сочиняют новые музыкальные произведения и даже изобретают новые блюда и рецепты.

В этой главе вы обучите модель машинного обучения распознавать жанр произведения искусства, основываясь только на его фотографии.

Вы не замечали, что постеры фильмов определенного жанра чем-то похожи друг на друга? Например, постеры триллеров в основном содержат крупные контрастные надписи, и для их дизайна используют темные цвета. Постеры романтических фильмов, наоборот, как правило, исполняют в светлых тонах, для их надписей используют утонченный декоративный шрифт. На постерах научно-фантастических фильмов часто можно увидеть изображения космических кораблей, звезд или планет.

Мы можем даже сами не замечать, как, едва бросив взгляд на постер, мы угадываем жанр фильма, даже не вчитываясь в то, что на нем написано (рис. 5.1).

А теперь вам предстоит научить компьютер распознавать общие черты у произведений искусства в определенном жанре. Например, найти что-то общее у постеров к боевикам. Или у обложек видеоигр про гонки. Или у обложек рэп-альбомов... Вы убедитесь в том, что компьютер легко справится и с задачей определять жанр произведения искусства, например книги, на основе только постера или иллюстрации на обложке.

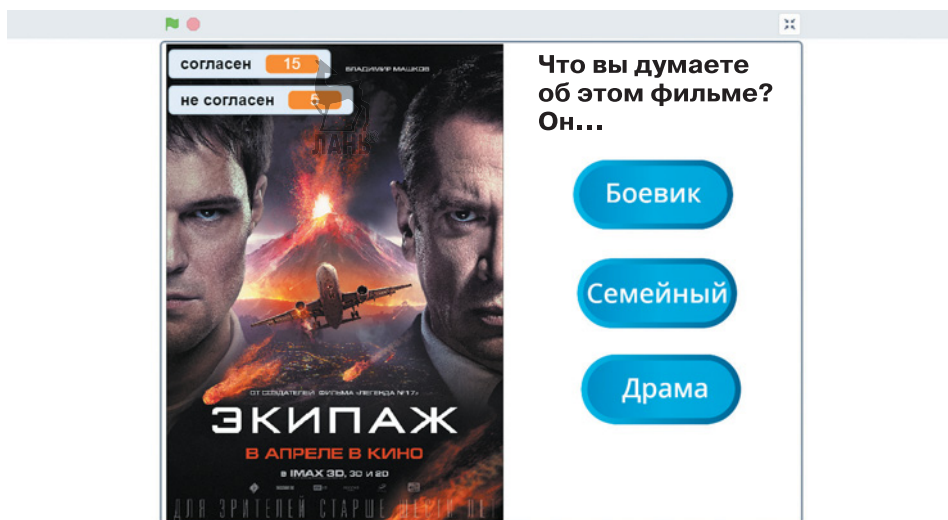


Рис. 5.1. Модель машинного обучения может научиться распознавать жанры кино

Давайте же начнем!

Выполнение проекта

Для начала выберите вид произведений искусства, представленный в разных жанрах и для рекламы которого создают обложки или постеры.

Например, вы можете выбрать:

- книги (могут распознаваться по иллюстрациям на обложке);
- фильмы (могут распознаваться по постерам);
- видеоигры (могут распознаваться по иллюстрациям на обложке или в артбуке);
- музыкальные альбомы (могут распознаваться по иллюстрациям на обложке альбома).

В качестве обучающих примеров для этого проекта вам нужно будет сделать подборку объектов выбранного вида. Проще всего это сделать, если найти веб-сайты, на которых книги, фильмы, видеоигры или музыкальные альбомы сгруппированы по жанрам. Например, если вы выбрали книги, можно найти много обучающих примеров на сайтах книжных магазинов и библиотек. Если вы выбрали музыкальные альбомы или видеоигры, в подборке обучающих примеров вам помогут сайты магазинов, где продаются такие товары, или стриминговые сервисы (например, «Яндекс.Музыка»).

После этого выберите несколько жанров, которые компьютер должен будет научиться распознавать. При этом учтите, что обучить компьютер будет гораздо легче, если выбрать сильно отличающиеся жанры. Например, компьютеру будет проще распознать разницу между постерами мелодрам и боевиков, чем между постерами боевиков и приключенческих фильмов.

На скриншотах этого проекта вы увидите модель машинного обучения, которую я обучил распознавать по постеру три жанра фильмов: боевик, для семейного просмотра (семейный) и драма.

Теперь, когда вы выбрали вид произведений искусства и жанры, вы можете приступить к обучению модели.

Обучение модели

1. Создайте новый проект машинного обучения и назовите его «Судить о книге по обложке». В качестве распознаваемого типа данных выберите «изображения».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 5.2.

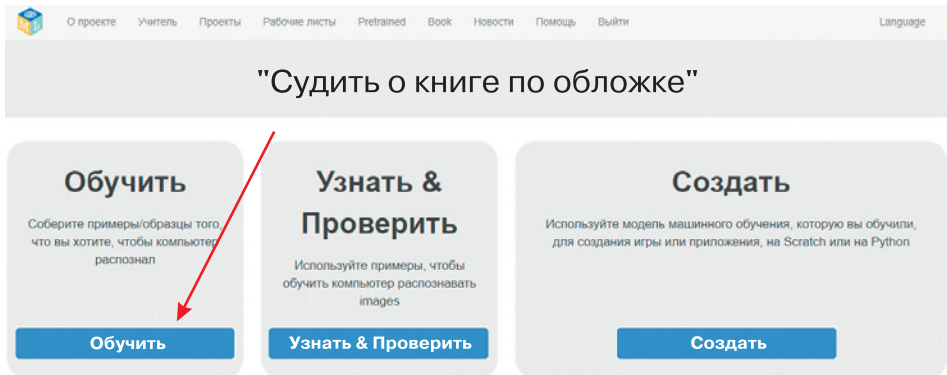


Рис. 5.2. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

3. Нажмите на кнопку «Добавить новую метку» (рис. 5.3) для создания первой коллекции и дайте ей имя одного из выбранных вами жанров.

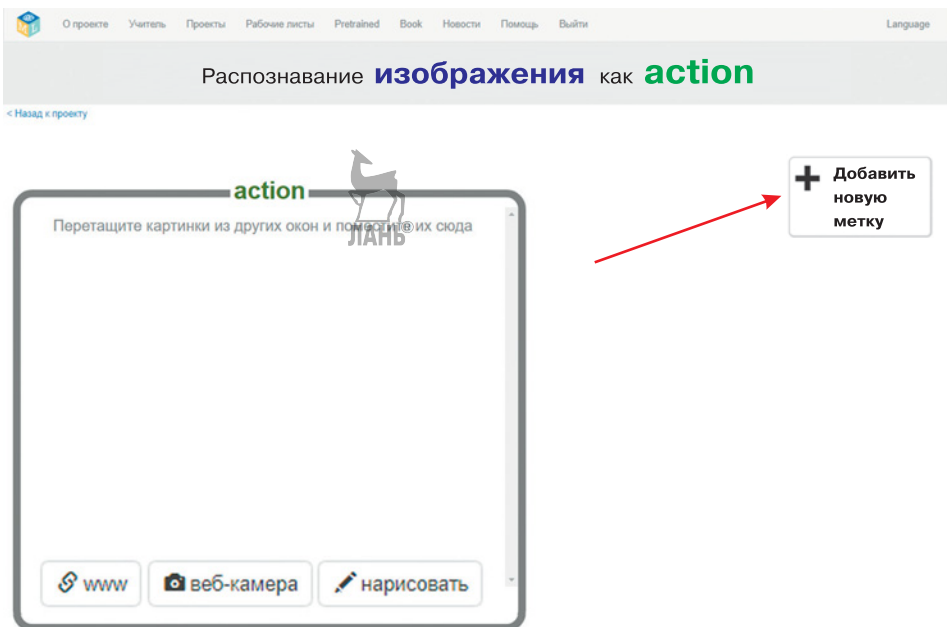


Рис. 5.3. Нажмите на кнопку «Добавить новую метку» для создания новой коллекции примеров*

* Слово «action» будет означать «боевик». — Прим. пер.

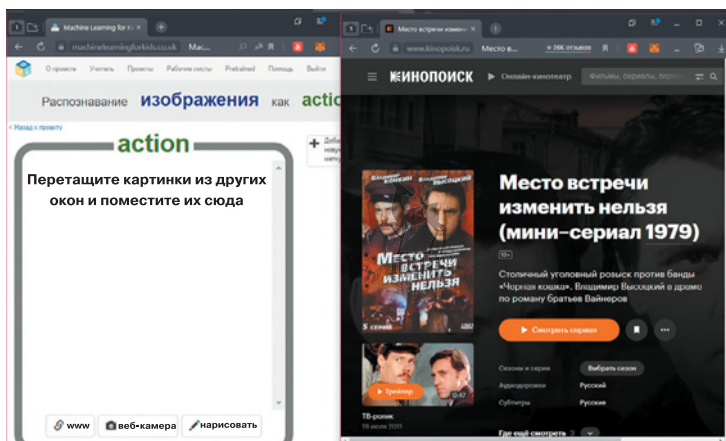


Рис. 5.4. Расположите два окна браузера рядом

4. Откройте второе окно вашего браузера (обычно в меню есть опция «Новое окно») и расположите эти два окна рядом, как показано на рисунке 5.4. Во втором окне начните поиск фотографий произведений искусства выбранного жанра. Вы можете воспользоваться сайтом «Кинопоиск» (рис. 5.5).

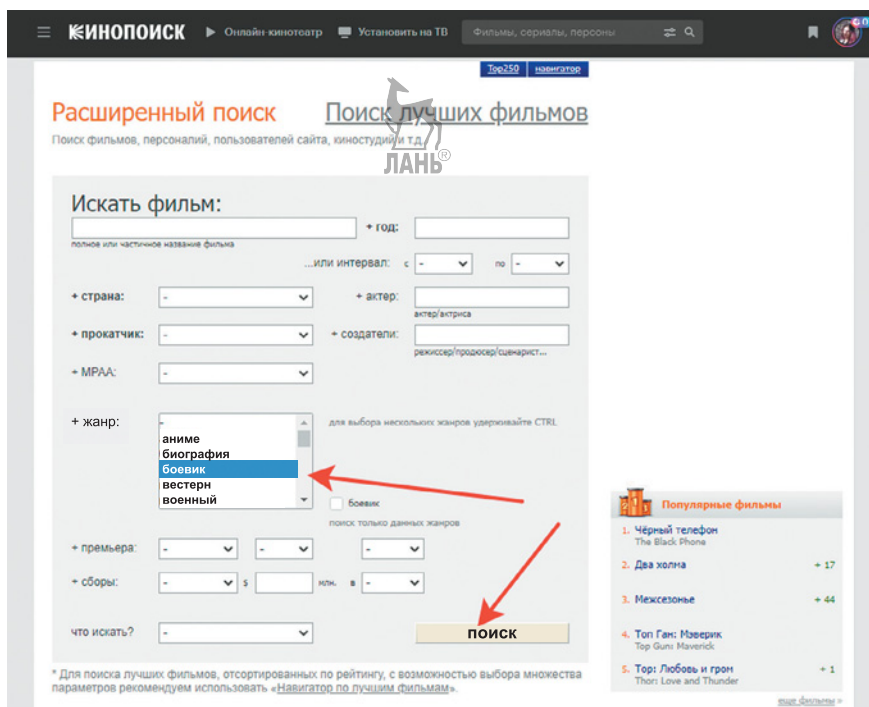


Рис. 5.5. Если вы используете «Кинопоиск», выберите нужный жанр в форме поиска

5. Перетащите подходящую фотографию (обложку книги, постер фильма, обложку видеоигры или музыкального альбома) из окна поиска прямо в коллекцию в окне проекта. Вы увидите, как в коллекции появилась уменьшенная версия этой фотографии (рис. 5.6). Если этого не произошло, попробуйте перетащить фотографию еще раз.

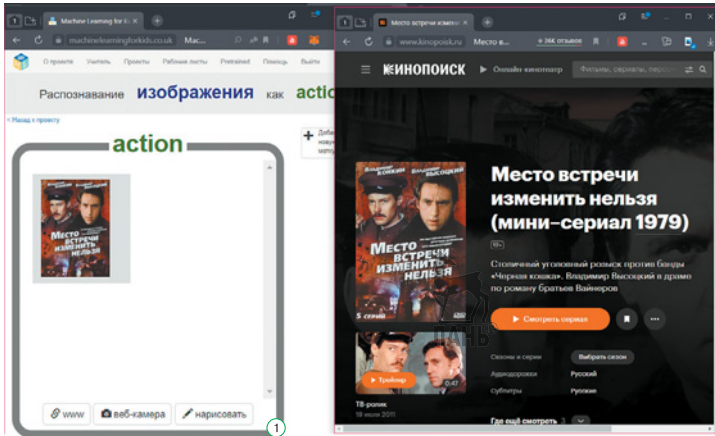


Рис. 5.6. Мой первый обучающий пример для распознавания боевиков

6. Повторяйте шаг 5 до тех пор, пока в коллекции не наберется как минимум 10 разных изображений произведений искусства выбранного вами жанра (рис. 5.7).

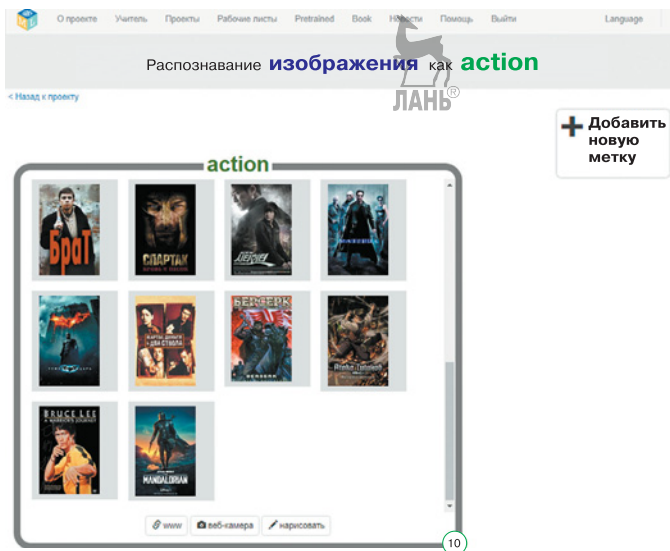


Рис. 5.7. Мои обучающие примеры для распознавания постеров боевиков

7. Повторите шаги с 3 по 6 для всех жанров, которые должна будет научиться распознавать ваша модель, как показано на рисунке 5.8.

Постарайтесь собрать примерно одинаковое количество обучающих примеров для каждого жанра, избегая того, чтобы одни коллекции были практически пустыми, а другие — избыточно наполненным.

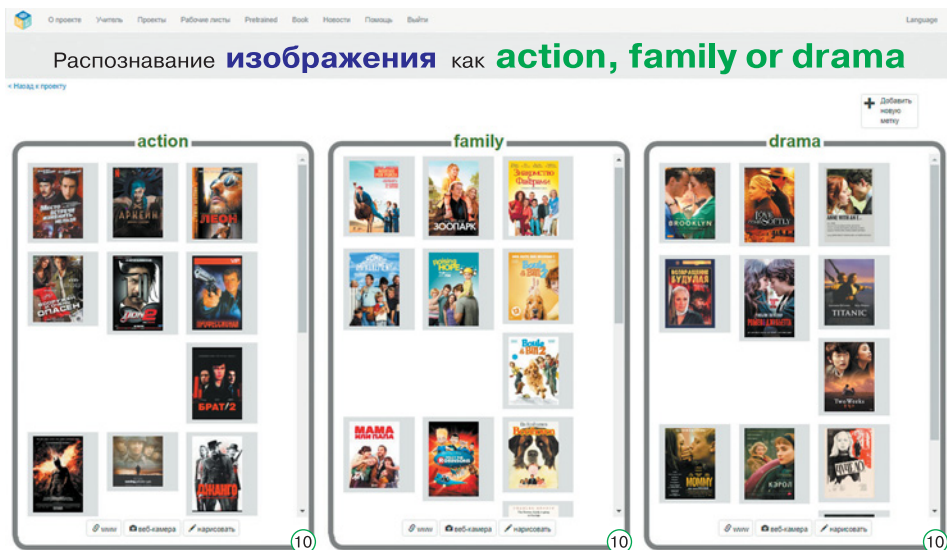


Рис. 5.8. Обучающие примеры для распознавания различных типов постеров к фильмам

8. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

9. Нажмите на кнопку «Узнать и проверить» (рис. 5.9), чтобы перейти к следующему этапу проекта.

10. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 5.10).

Компьютер будет использовать собранные вами обучающие примеры для поиска общих закономерностей на обложках или постерах. Обучение модели может занять несколько минут, но вы, пока ждете, можете перейти к следующему шагу и выполнить его во втором окне браузера.

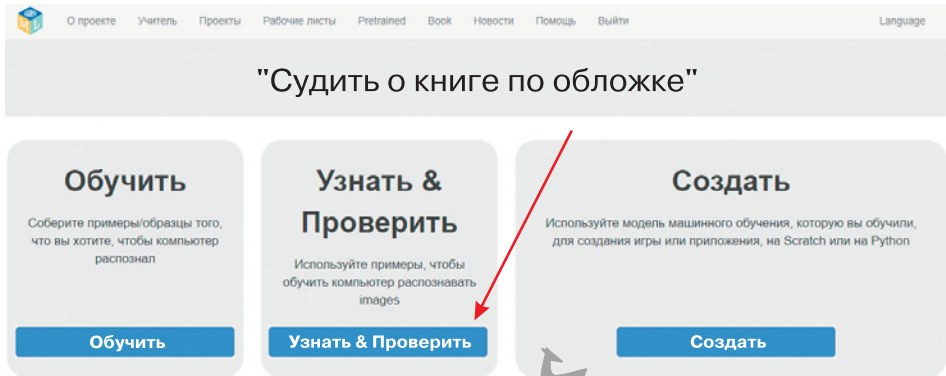


Рис. 5.9. Переход ко второму этапу работы над проектом — этапу «Узнать и проверить»

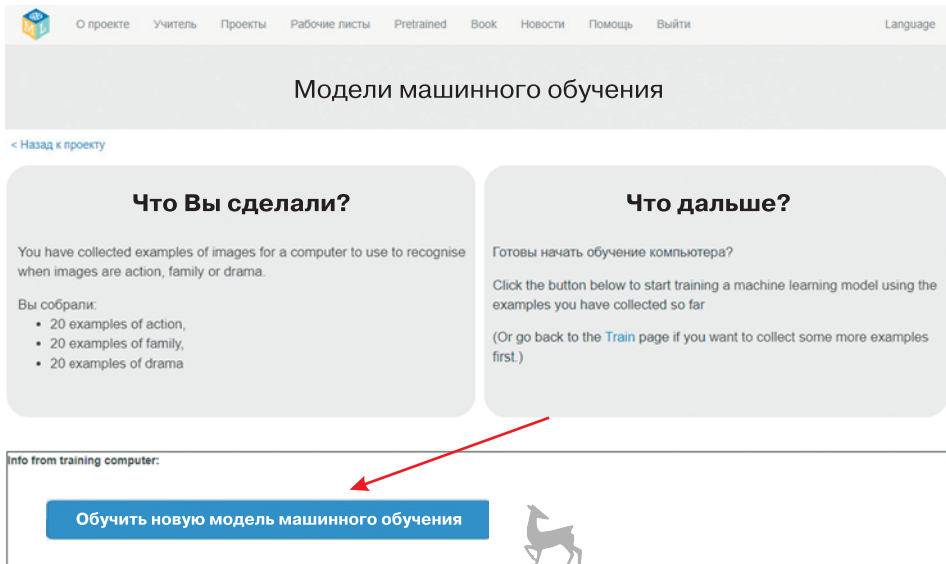


Рис. 5.10. Нажмите на кнопку «Обучить новую модель машинного обучения», чтобы начать обучение

Подготовка проекта

Теперь нужно проверить, способна ли ваша модель машинного обучения распознать жанр произведения искусства по изображению, которое она раньше не видела. Чтобы протестировать модель, вам нужно будет скачать несколько новых изображений, которые вы не использовали в качестве обучающих примеров.

Затем вы создадите скрипт Scratch, который с помощью этих изображений поможет проверить, насколько хорошо работает ваша модель.

1. Найдите еще несколько изображений для каждого жанра, который вы выбрали, и сохраните их на свой компьютер. Чтобы сохранить фотографию, кликните по ней правой кнопкой мыши и выберите «Сохранить изображение» или «Сохранить изображение как...» (рис. 5.11).



Примечание

Не используйте те же фотографии, что использовали в качестве примеров для обучения модели. Вам нужно проверить, насколько хорошо компьютер научился распознавать изображения, а не насколько хорошо он научился их запоминать.

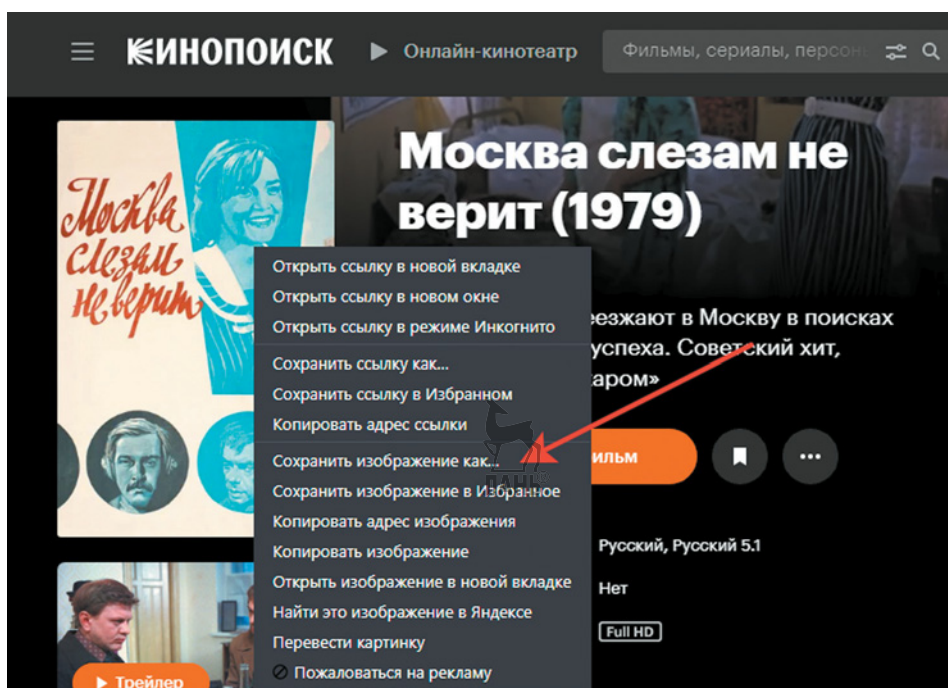


Рис. 5.11. Сохранение изображения на компьютер

Сохраните эти изображения в отдельную папку на своем компьютере, как показано на рисунке 5.12. Чем больше изображений вы сохраните, тем качественнее протестируете свою модель.

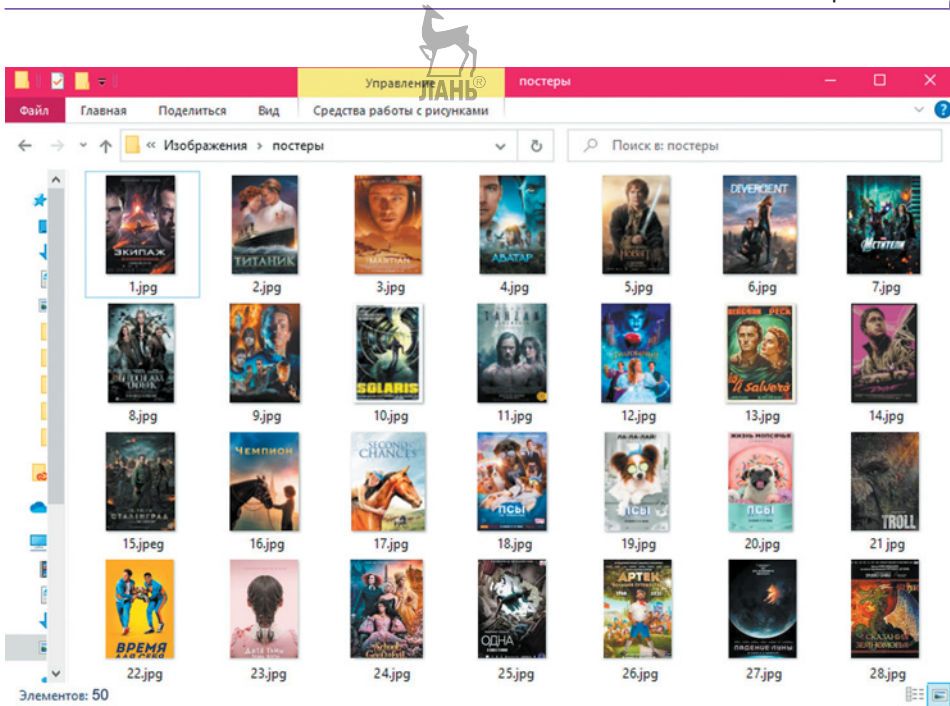


Рис. 5.12. Подготовка изображений — тестовых примеров

2. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

3. Нажмите на кнопку «Создать» (рис. 5.13).

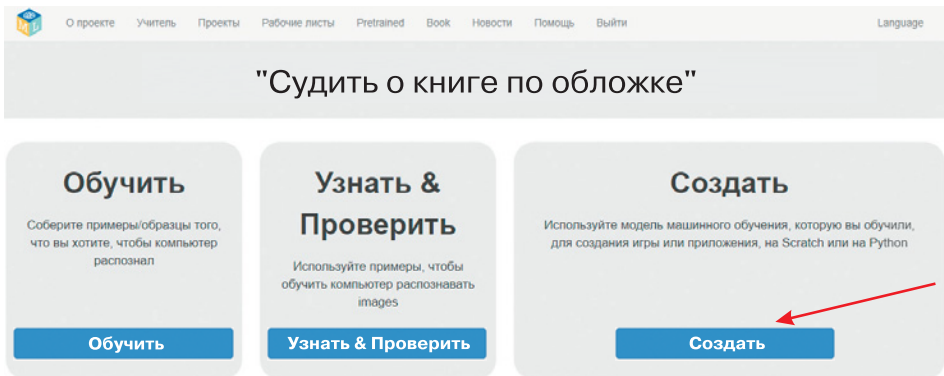


Рис. 5.13. Переход к третьему этапу проекта — этапу «Создать»

4. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch.

5. Нажмите на спрайт кота (Спрайт 1) на панели спрайтов в правом нижнем углу экрана. Затем перейдите на вкладку «Костюмы» в левом верхнем углу.

6. Наведите указатель мыши на иконку «Выбрать костюм» в левом нижнем углу экрана. Нажмите на кнопку «Загрузить костюм» и найдите папку, в которую вы загрузили тестовые примеры.

7. Выделите все изображения, которые вы сохранили в эту папку на шаге 1, чтобы одновременно загрузить их все как костюмы для одного спрайта.



Примечание

Убедитесь, что вы не загружаете изображения для отдельных спрайтов. Вы должны видеть только один спрайт в правом нижнем углу экрана, но много костюмов к нему на панели слева.

8. Измените имя спрайта кота. Для этого найдите текстовое поле Спрайта и введите в него название «Обучающие примеры», как показано на рисунке 5.14.

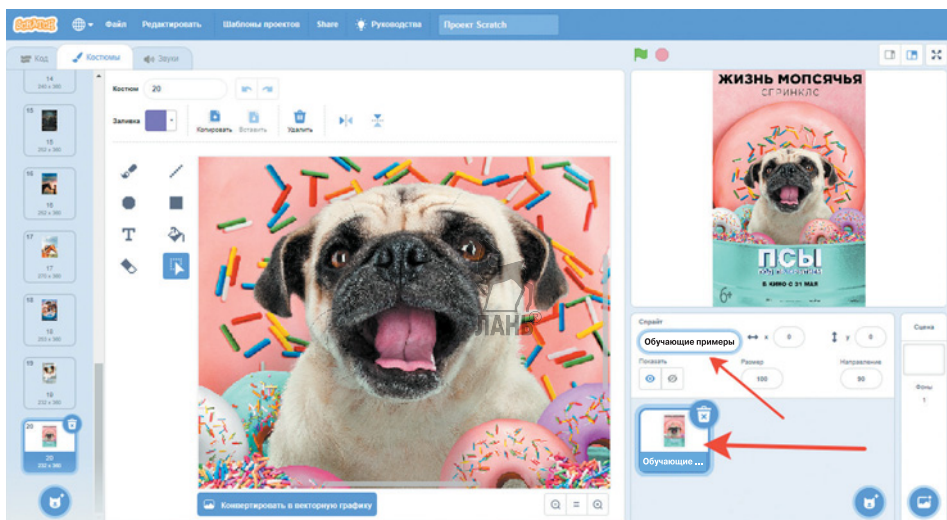


Рис. 5.14. Создание спрайта для хранения тестовых изображений

9. Для этого проекта вам также понадобятся спрайты кнопок. Наведите указатель мыши на иконку «Выбрать спрайт» в правом нижнем углу экрана.

Если вы хотите нарисовать свои собственные кнопки, нажмите на кнопку «Нарисовать». При этом не волнуйтесь, если вдруг при рисовании у вас что-то не будет получаться с первого раза, вы в любой момент можете нажать на синюю стрелочку «Отме-

нить» рядом с именем костюма и вернуться на несколько шагов назад. Не унывайте!

Если же вы не хотите рисовать, воспользуйтесь готовыми шаблонами из библиотеки Scratch, нажав на кнопку «Выбрать спрайт» (рис. 5.15). Создайте по одной кнопке для каждого из жанров.

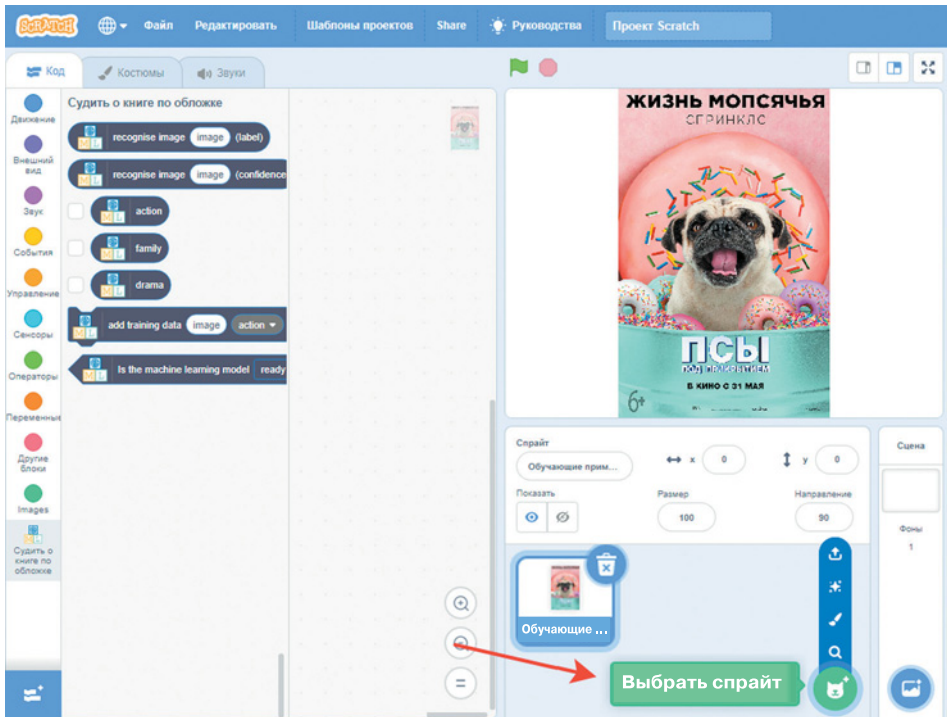


Рис. 5.15. Нажмите на кнопку «Выбрать спрайт», чтобы получить доступ к библиотеке спрайтов Scratch

10. Переименуйте спрайты кнопок так, чтобы их имена соответствовали названиям жанров, как показано на рисунке 5.16. У меня это три кнопки с именами «Боевик», «Семейный» и «Драма».

11. Чтобы добавить надписи на кнопки, перейдите на вкладку «Костюмы» и выберите инструмент «Текст» (обозначен буквой Т). Если захотите изменить цвета надписей — используйте инструмент «Заливка» (рис. 5.17). Сделайте так, чтобы надписи на кнопках соответствовали названиям жанров.

12. Далее вам нужно создать три новые переменные. Перейдите на вкладку «Код», выберите категорию «Переменные» в Панели инструментов и нажмите на кнопку «Создать переменную» (рис. 5.18).

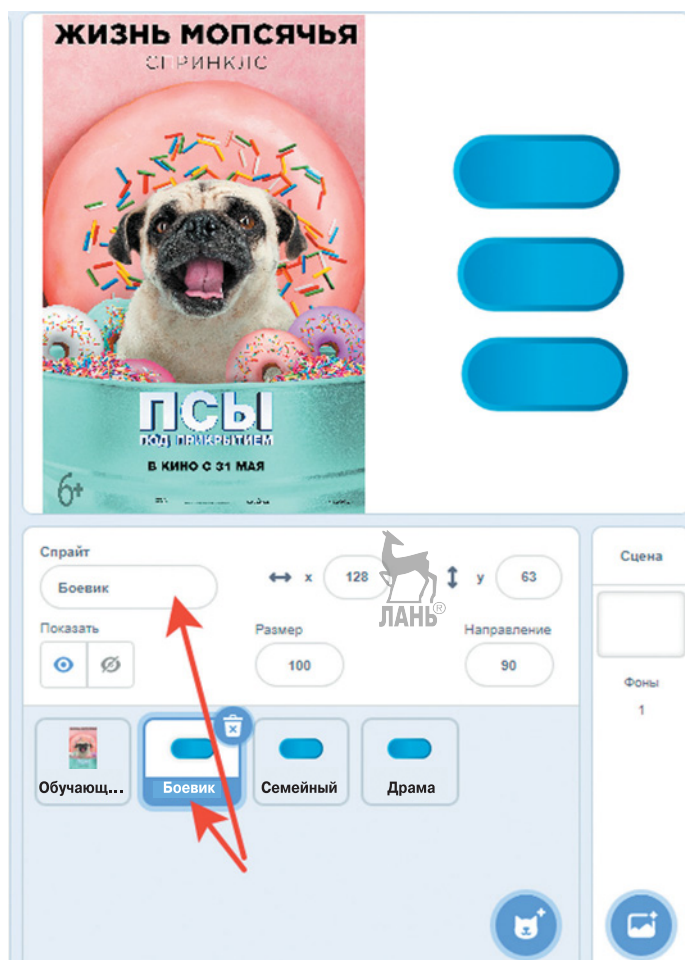


Рис. 5.16. Создайте кнопки для каждого из жанров



Примечание

Для всех трех переменных выберите опцию «Для всех спрайтов».



Две из этих переменных будут подсчитывать, сколько раз вы согласились или не согласились с решением компьютера, т. е. угадал компьютер жанр или не угадал. Дайте первой переменной имя «согласен», а второй — «не согласен».

Третья переменная будет хранить имя жанра, который компьютер выбрал для последнего рассмотренного изображения. Назовите эту переменную «компьютер».

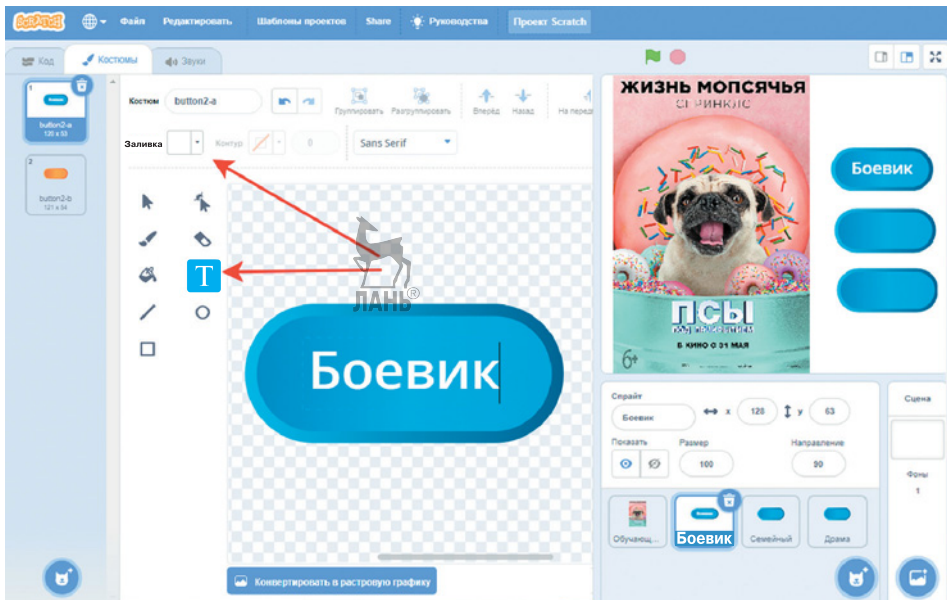


Рис. 5.17. Я использовал инструменты «Текст» и «Заливка», чтобы добавить на каждую из кнопок белую надпись с названием жанра

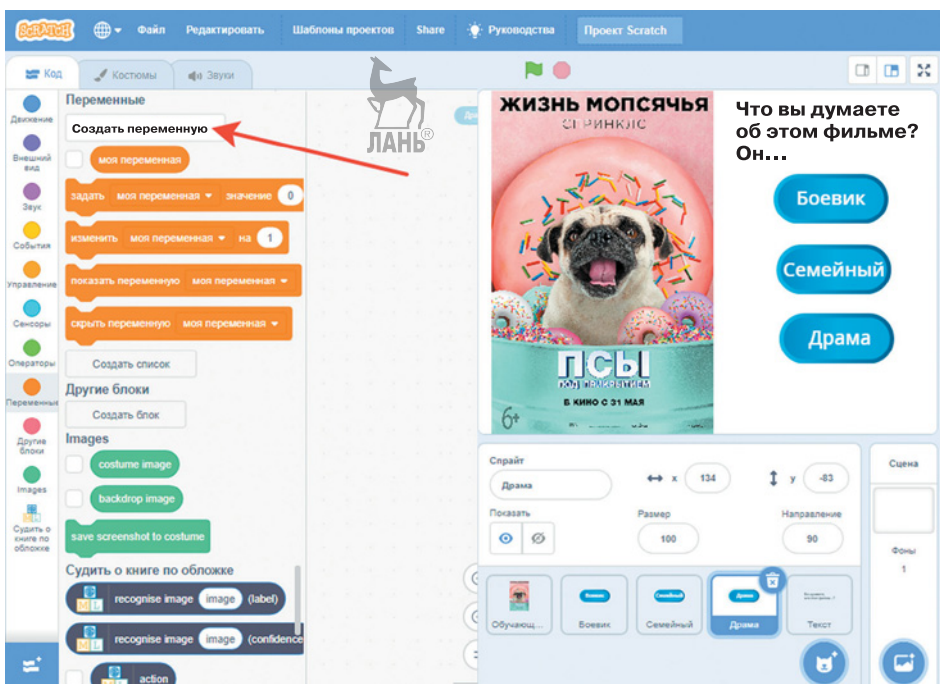


Рис. 5.18. Нажмите на кнопку «Создать переменную», чтобы добавить в проект три переменные

13. Убедитесь, что в чекбоксах рядом с переменными «согласен» и «не согласен» стоят галочки. В этом случае они отображаются на Сцене, и вы сможете видеть результаты прямо во время тестирования проекта. А галочку в чекбоксе рядом с переменной «компьютер», наоборот, нужно снять.

14. Нажмите на спрайт «Обучающие примеры». Это тот спрайт, для которого вы добавляли костюмы — тестовые изображения.

15. Создайте такой же скрипт, как показан на рисунке 5.19.



Примечание

Если вы не знаете, как создаются скрипты в Scratch, возвратитесь к разделу «Программирование в среде Scratch» на с. 15.

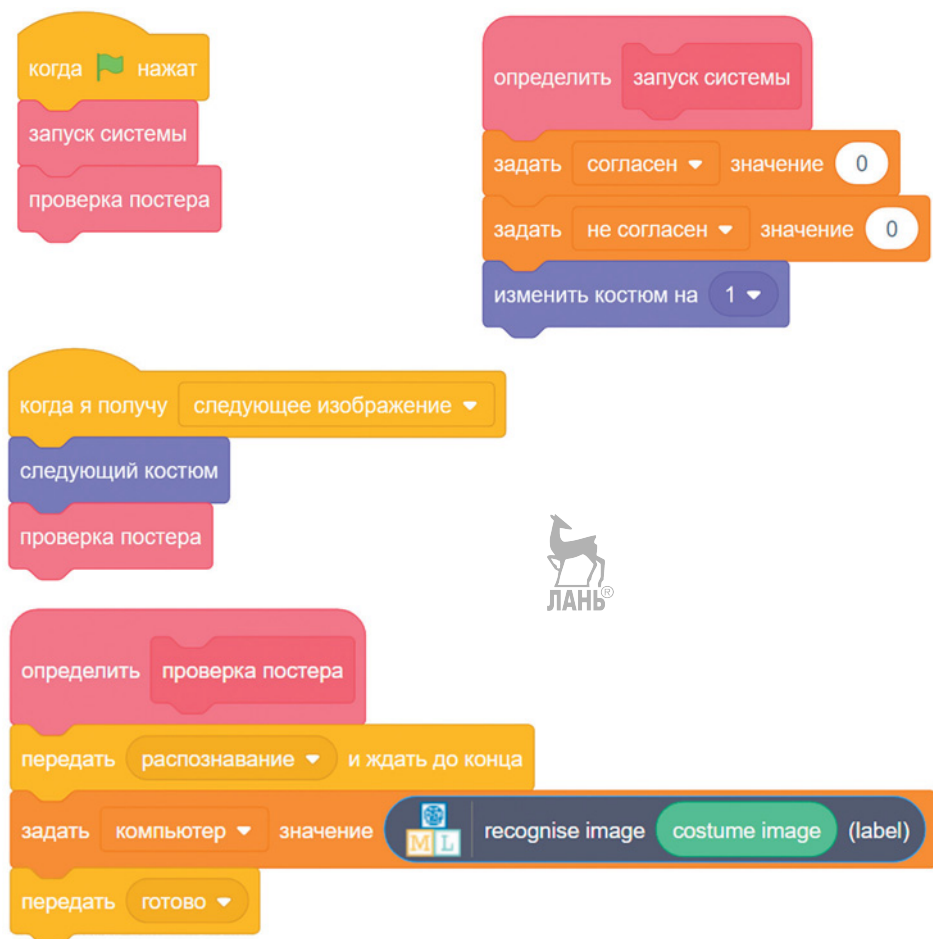


Рис. 5.19. Скрипт для распознавания постеров фильмов

В блоке «изменить костюм на» используйте выпадающий список, чтобы изменить костюм на ваше первое тестовое изображение, как показано на рисунке 5.19 (мое первое тестовое изображение имеет имя «1»).

16. Нажмите на первую кнопку с названием жанра (рис. 5.20).

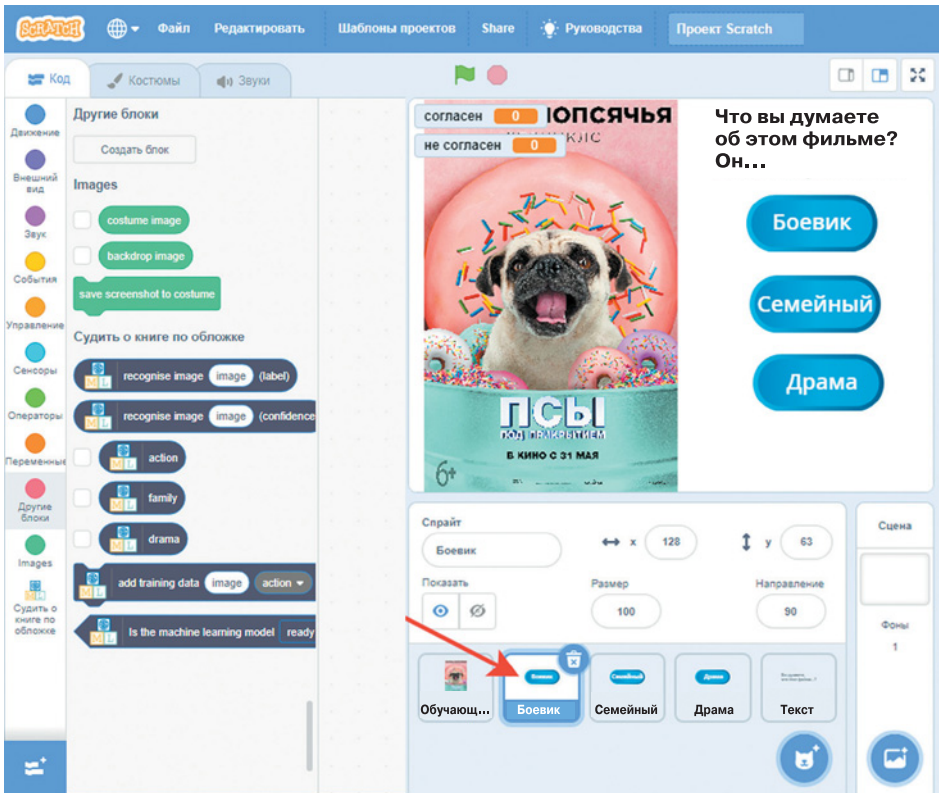


Рис. 5.20. Кнопки с названиями жанров

17. Создайте такой же скрипт, как показан на рисунке 5.21. Измените блок с названием категории «action» таким образом, чтобы он соответствовал названию первой кнопки с названием жанра. Помните, что программа понимает только надписи, сделанные латиницей.

Компьютер будет использовать этот скрипт, когда пользователь нажмет на кнопку, чтобы угадать жанр. Если выбор пользователя будет соответствовать тому, что распознала модель, счетчик переменной «согласен» увеличится на 1. Иначе, если выбор пользователя и компьютера не совпадет, увеличится счетчик переменной «не согласен».

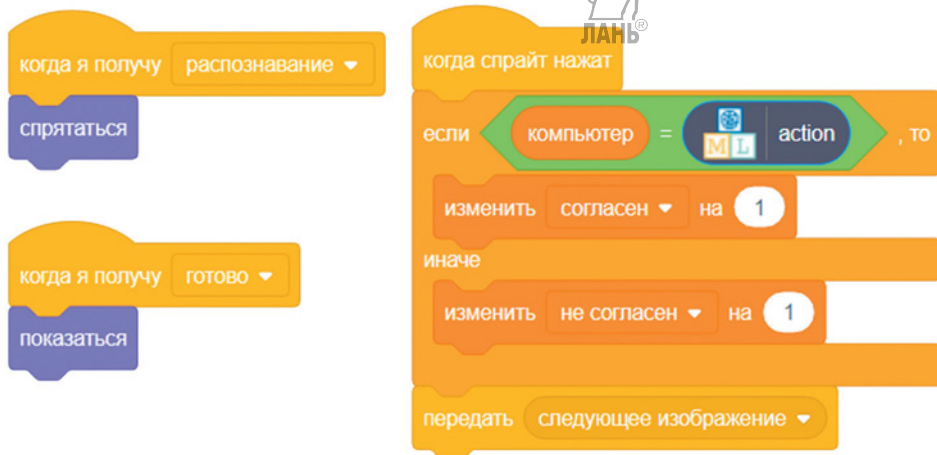


Рис. 5.21. Скрипт для первой кнопки с названием жанра

18. Нажмите на спрайт следующей кнопки с названием жанра. Создайте для нее такой же скрипт, как в шаге 17, но измените название категории, как показано на рисунке 5.22 (как и в предыдущем шаге, это название должно соответствовать названию жанра, но на этот раз — второй кнопки). В моем примере вторая кнопка создана для фильмов для семейного просмотра.

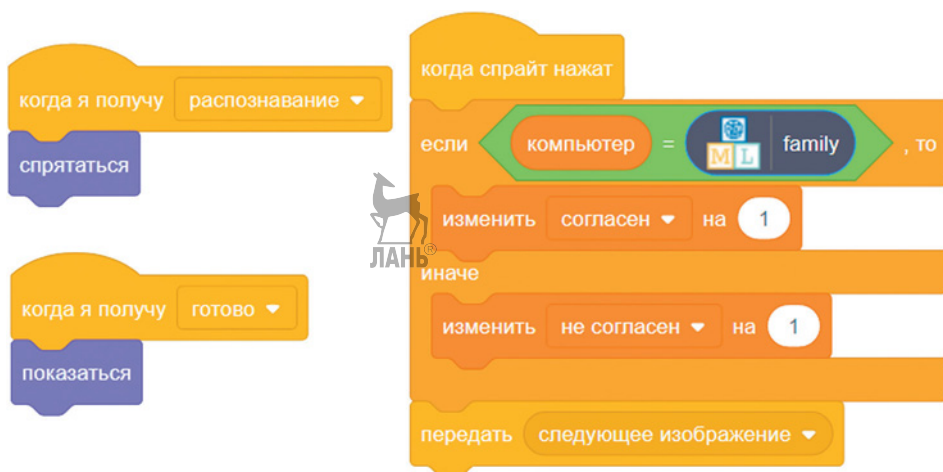


Рис. 5.22. Скрипт для второй кнопки с названием жанра

19. Повторяйте шаг 17 до тех пор, пока каждая из ваших кнопок не будет иметь такой же скрипт (рис. 5.23).

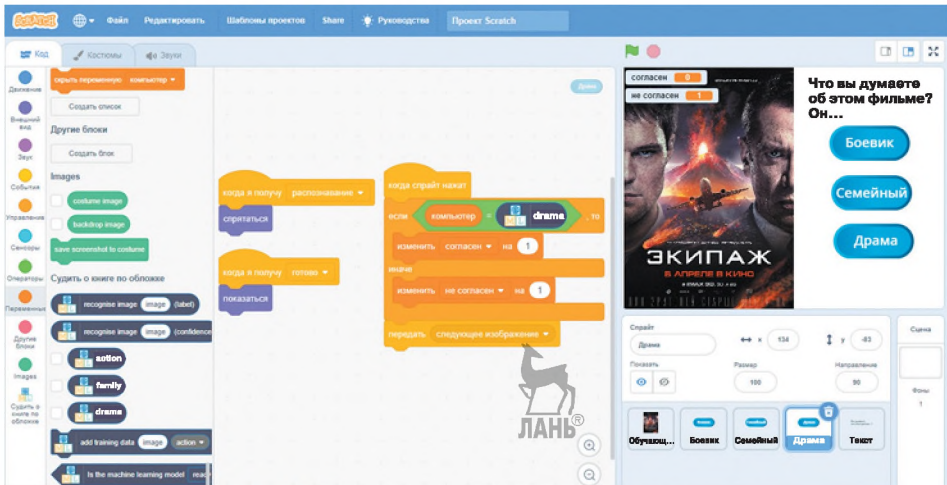


Рис. 5.23. Каждая кнопка с названием жанра должна иметь свой скрипт

Готово! Наконец настало время запустить и протестировать вашу модель машинного обучения!

Тестирование модели

Чтобы эксперимент по-настоящему удался, пригласите кого-нибудь другого протестировать ваш проект. Будет лучше, если тестировать модель будет человек, который еще не видел подготовленных вами тестовых изображений.

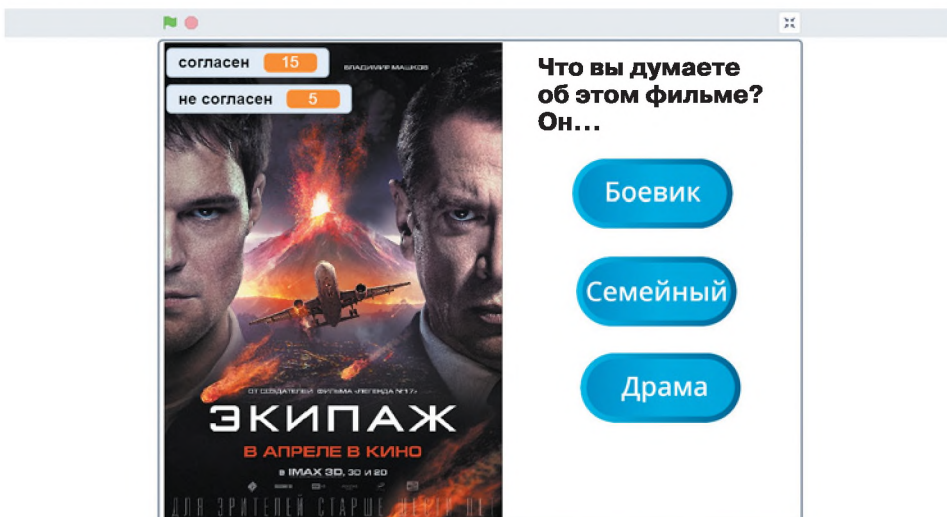


Рис. 5.24. Тестирование модели машинного обучения

Как только будет нажат зеленый флажок, Scratch покажет на экране тестовое изображение и попросит пользователя определить, к какому жанру относится это произведение (и нажать на соответствующую кнопку). Ваш скрипт будет вести подсчет результатов — сколько раз выбор пользователя совпал с выбором модели машинного обучения, а сколько — не совпал (рис. 5.24).

Попросите пользователя, который тестирует ваш проект, стараться выбирать жанр только на основании изображения, даже если он знаком с этим произведением.

Обзор и улучшение проекта



В этом проекте вы обучили модель машинного обучения распознаванию стилей, характерных для произведений искусства разных жанров.

Если же ваша модель справляется не очень хорошо и значения счетчиков показывают, что компьютер был чаще не согласен с пользователем, чем согласен, вернитесь к этапу обучения модели и добавьте больше примеров в каждую из категорий. После этого обучите обновленную модель и попробуйте протестировать ее снова. В целом же не забывайте наш главный принцип — чем больше обучающих примеров использует модель, тем лучше результаты ее работы.

Что вы узнали

В этой главе вы обучили еще одну модель машинного обучения распознавать изображения. Но если в предыдущих двух главах вы обучали свои модели распознаванию объектов на изображениях, то в этом — распознаванию стилей изображений.

Вы также узнали, что одним из способов определения эффективности системы машинного обучения является сравнение ответов, которые она дает, с ответами, которые люди дали на эти же вопросы.

В следующей главе вам предстоит узнать о еще одном способе применения систем распознавания изображений — распознавание рукописных текстов.



ГЛАВА 6

Сортировка писем



В предыдущих главах вы выполняли проекты, связанные с обучением компьютера распознаванию изображений. Есть много полезной работы, которую нам могут помочь выполнить компьютеры, если они умеют «видеть». Одной из таких задач является технология **оптического распознавания символов** (OCR, аббревиатура от англ.: Optical Character Recognition), позволяющая электронной машине распознавать буквы и цифры.

Компьютер, обученный оптическому распознаванию символов путем изучения большого количества примеров написания этих символов, может, например, читать вслух (т. е. озвучивать) тексты газет и книг.

Оптическое распознавание символов используется вместе с системами преобразования текста в речь, которые могут озвучивать распознанные слова. В таком сочетании они могут помочь людям с нарушением зрения читать текст, который те не могут увидеть сами.

Историки, библиотекари и архивариусы используют оптическое распознавание символов для изучения исторических документов. Машинное обучение может распознавать слова и позволяет выполнять поиск по публикациям, которым сотни лет.

Оптическое распознавание символов используется и на дорогах — для анализа номерных знаков автомобилей. Системы автоматического распознавания номерных знаков используются для обеспечения непрерывности дорожного движения. Они позволяют автоматически взимать плату за проезд по платным трассам, а также обеспечивать безопасность на дорогах, распознавая автомобили, которые движутся с превышением скорости.

Многие компании используют оптическое распознавание символов для обработки различных анкет и документов. Если вы заполняете анкету или выписываете чек, системы машинного обучения часто используют оптическое распознавание символов, чтобы понять, что же вы написали.

Если вы путешествовали за границу, то, возможно, знакомы с приложениями для смартфона, которые выводят на экран перевод слов с вывески или меню на ваш родной язык, когда вы просто наводите камеру устройства на иностранный текст. Такие приложения также используют оптическое распознавание символов.

Одним из самых распространенных способов использования оптического распознавания символов является сортировка почты, о чем и пойдет речь в этой главе. Вы научите компьютер распознавать рукописный текст и увидите, как оптическое распознавание символов помогает быстро сортировать письма. Вы создадите проект Scratch — почтовый офис, который сможет автоматически сортировать письма благодаря распознаванию указанной на конверте аббревиатуре города адресата (рис. 6.1).

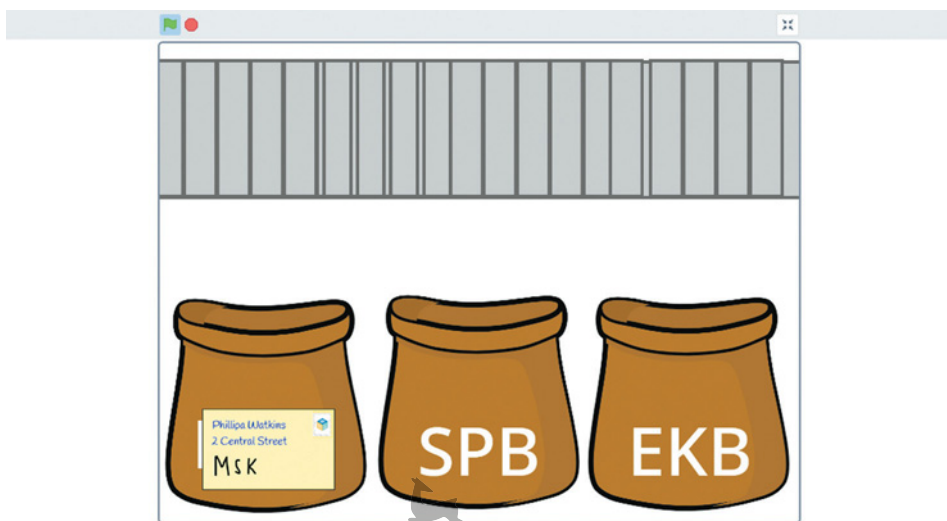


Рис. 6.1. Сортировка писем через распознавание аббревиатуры города

Давайте же начнем!

Выполнение проекта

Для начала выберите три больших города, принятые сокращения названий которых должна будет распознавать ваша модель. Например, для этого проекта можно выбрать Москву, Санкт-Петербург и Екатеринбург.

Для этих городов принято использовать следующие аббревиатуры:

- МСК (MSK) — для Москвы;

- СПБ (SPB) — для Санкт-Петербурга;
- ЕКБ (ЕКВ) — для Екатеринбурга.

Итак, вам нужно выбрать три города и три разных кода, которые будут их обозначать.

Обучение модели

Для того чтобы обучить компьютер распознавать аббревиатуры, которые вы выбрали, вам нужно будет написать от руки примеры этих сокращений, а затем использовать их для обучения модели.

1. Создайте новый проект машинного обучения и назовите его «Почтовый офис». В качестве распознаваемого типа данных выберите «изображения».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 6.2.

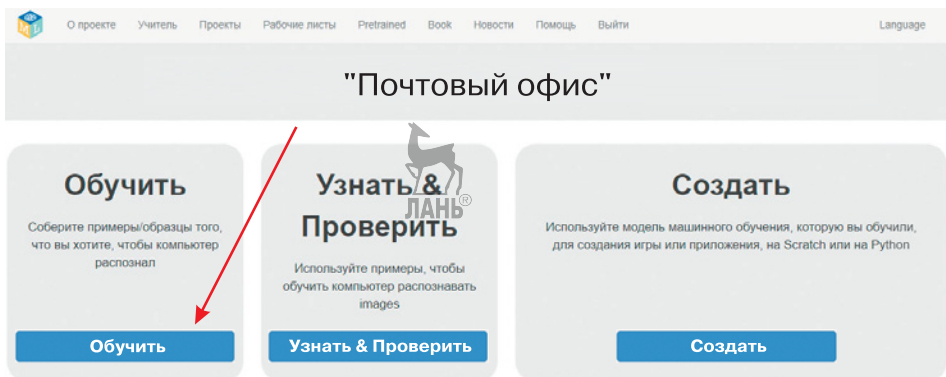


Рис. 6.2. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

3. Нажмите на кнопку «Добавить новую метку» (рис. 6.3) для создания первой коллекции и дайте ей имя в виде сокращения одного из выбранных вами городов.

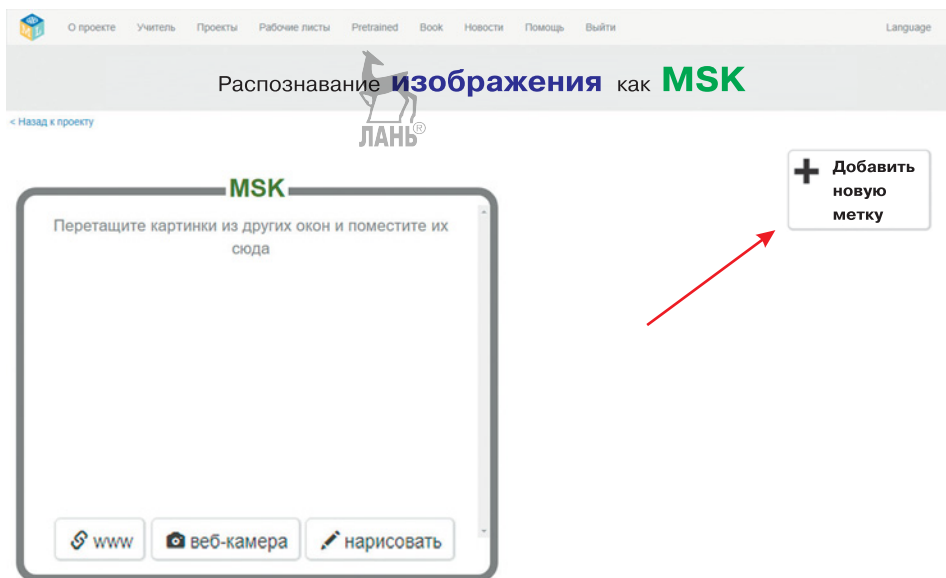


Рис. 6.3. Нажмите на кнопку «Добавить новую метку» для создания новой коллекции примеров

Воспользуйтесь инструментом «Нарисовать» в нижней части коллекции и изобразите аббревиатуру города Москвы, как показано на рисунке 6.4. Эту картинку должен будет научиться распознавать компьютер для выбранного города. Когда закончите, нажмите на кнопку «Добавить», чтобы сохранить результат.

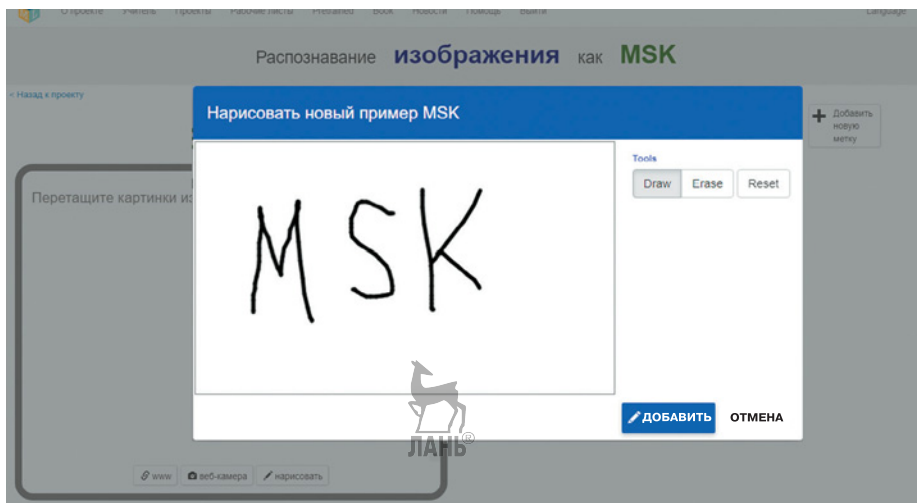


Рис. 6.4. Нажмите на кнопку «Нарисовать», чтобы добавить обучающие примеры для этого проекта



Примечание

Гораздо легче нарисовать аббревиатуру, если у вас есть устройство с тачскрином или графический планшет, но даже если нет, вы можете рисовать с помощью мыши. Не волнуйтесь: в этом проекте совершенно неважно, насколько аккуратно у вас получилось вывести нужные символы.

4. Повторяйте шаг 4 до тех пор, пока в коллекции не наберется хотя бы 10 обучающих примеров, как показано на рисунке 6.5.

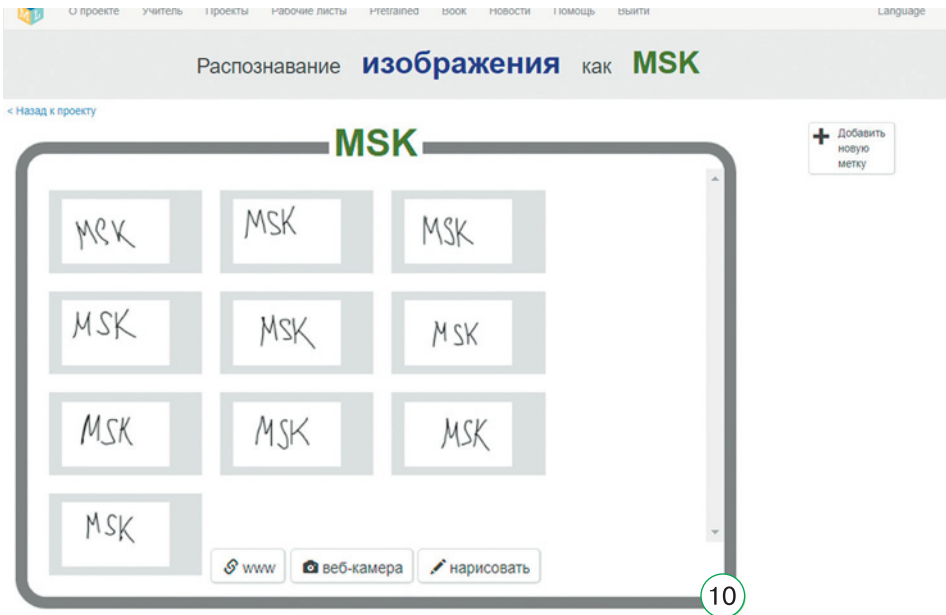


Рис. 6.5. Обучающие примеры для распознавания аббревиатуры Москвы

5. Повторяйте шаги 3–5 для остальных двух городов до тех пор, пока в каждой коллекции не будет собрано по меньшей мере 10 обучающих примеров, как показано на рисунке 6.6.

6. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

7. Нажмите на кнопку «Узнать и проверить» (рис. 6.7), чтобы перейти к следующему этапу проекта.



Рис. 6.6. Обучающие примеры для распознавания аббревиатур для всех трех городов

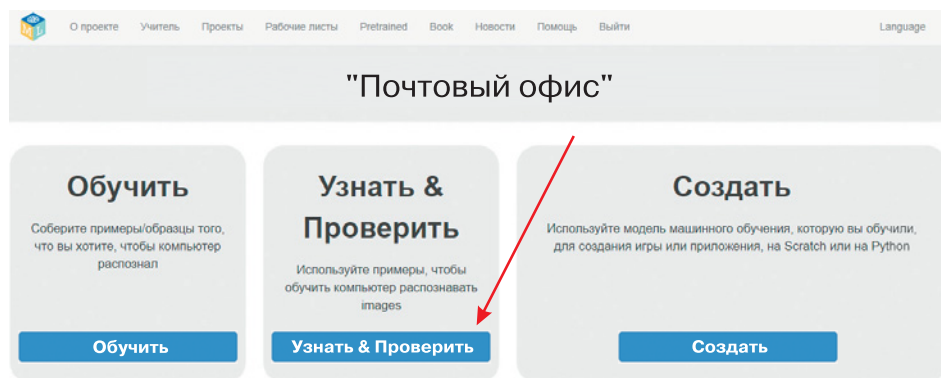


Рис. 6.7. Переход ко второму этапу работы над проектом — этапу «Узнать и проверить»

8. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 6.8). Компьютер будет использовать ваши рукописные примеры, чтобы научиться распознавать аббревиатуры разных городов. Но, поскольку все ваши обучающие примеры были написаны одной рукой, одним почерком и одним цветом, компьютер, скорее всего, справится с этой задачей.

Процесс обучения новой модели может занять несколько минут.

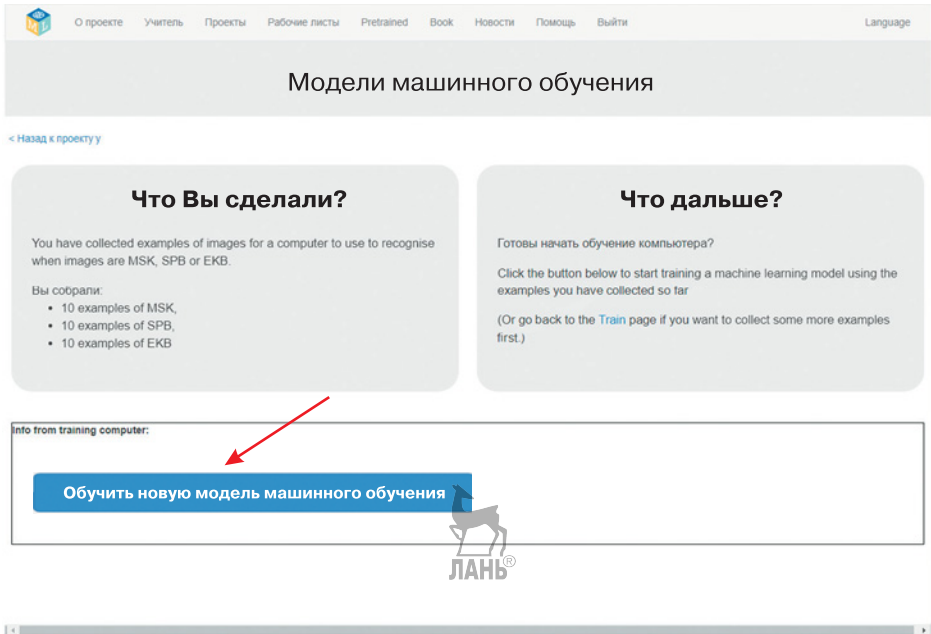


Рис. 6.8. Обучение новой модели машинного обучения распознаванию аббревиатуры

9. Настало время протестировать вашу модель машинного обучения! В предыдущих главах вы делали это, переходя в Scratch и создавая проекты, которые позволяли проверить, насколько хорошо компьютер научился распознавать и сортировать изображения, загруженные из Интернета или снятые веб-камерой. В этом проекте мы сначала протестируем модель прямо здесь и только после того, как убедимся, что она хорошо работает, перейдем в Scratch.



Примечание

Когда ваш учитель объясняет вам новую тему, он часто тестирует вас, чтобы определить, насколько хорошо вы ее поняли. Тестирование важно и в проектах машинного обучения, ведь сразу после обучения новой модели невозможно сказать, насколько хорошо она работает.

Для того чтобы протестировать вашу модель машинного обучения, нажмите на кнопку «Test by drawing» (в переводе с англ.: протестировать путем рисования), как показано на рисун-

Что Вы сделали?

You have trained a machine learning model to recognise when images are MSK, SPB or EKB.

You created the model on Friday, July 22, 2022 8:52 PM.

You have collected:

- 10 examples of MSK,
- 10 examples of SPB,
- 10 examples of EKB



Что дальше?

Try testing the machine learning model below. Enter an example image below, that you didn't include in the examples you used to train it. It will tell you what it recognises it as, and how confident it is in that.

If the computer seems to have learned to recognise things correctly, then you can go to Scratch and use what the computer has learned to make a game!

If the computer is getting too many things wrong, you might want to go back to the [Train](#) page and collect some more examples

Once you've done that, click on the button below to train a new machine learning model and see what difference the extra examples will make!

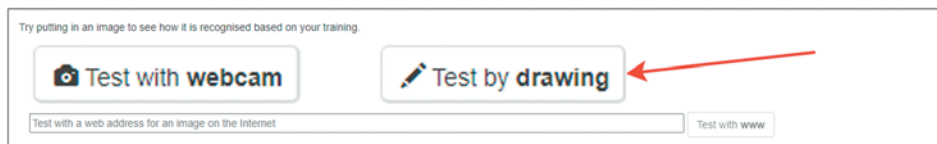


Рис. 6.9. Тестирование — это важная часть проектов машинного обучения

ке 6.9. Сделайте еще несколько вариантов записи аббревиатуры выбранных вами городов и проверьте, насколько хорошо ваша модель научилась распознавать рукописный текст.

Если вы не видите кнопку «Test by drawing», скорее всего, ваша модель еще не закончила обучение. Подождите еще пару минут. Помните притчу про зайца и черепаху?

Если же вас не устраивает, насколько хорошо ваша модель распознает надписи сокращений на тестовых примерах, вернитесь к этапу обучения и добавьте больше обучающих примеров в каждую коллекцию. Чем больше обучающих примеров вы соберете, тем лучше будет работать ваша модель. Главное, не забудьте после этого снова нажать на кнопку «Обучить новую модель машинного обучения».

Подготовка проекта

Теперь вы протестируете свою модель машинного обучения, создав виртуальный почтовый офис в Scratch, который будет использовать вашу систему оптического распознавания символов для сортировки конвертов.

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

2. Нажмите на кнопку «Создать» (рис. 6.10).

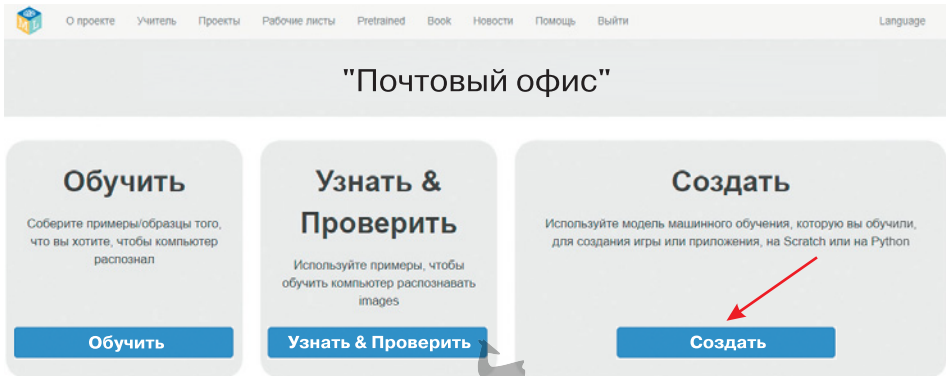


Рис. 6.10. Переход к третьему этапу проекта — этапу «Создать»

3. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch. Вы увидите новую категорию блоков на Панели инструментов (рис. 6.11), в которой содержатся блоки для работы с вашим проектом машинного обучения.

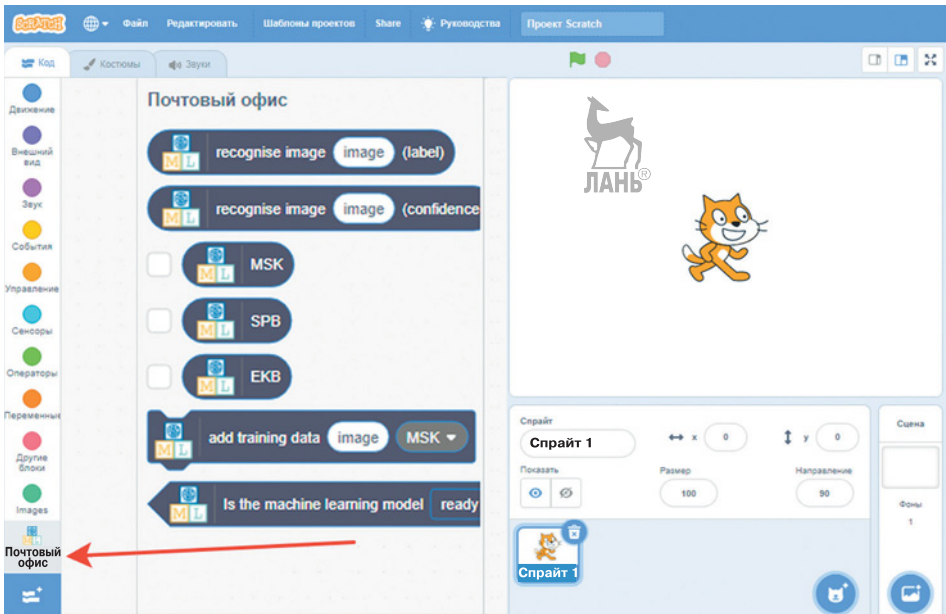


Рис. 6.11. Блоки для работы с вашим проектом машинного обучения в Scratch 3

4. В Панели инструментов Главного меню выберите **Шаблоны проектов** (рис. 6.12).

Здесь вы можете выбрать шаблоны проектов с частично написанными скриптами для экономии вашего времени.

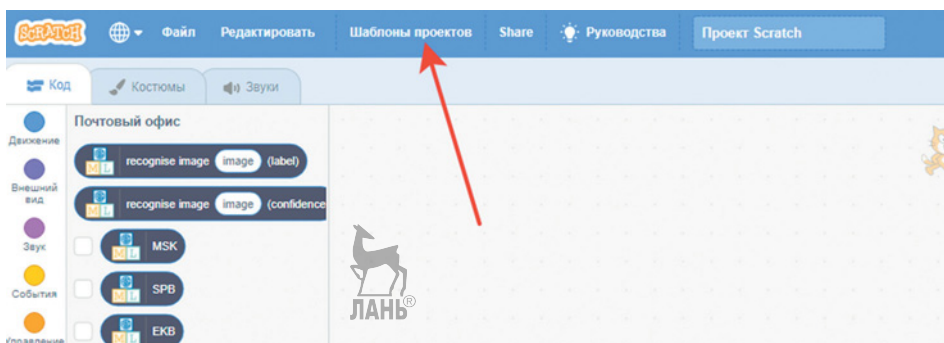


Рис. 6.12. Выберите пункт **Шаблоны проектов** в Главном меню Scratch

5. Найдите и выберите шаблон «Sorting Office», как показано на рисунке 6.13.

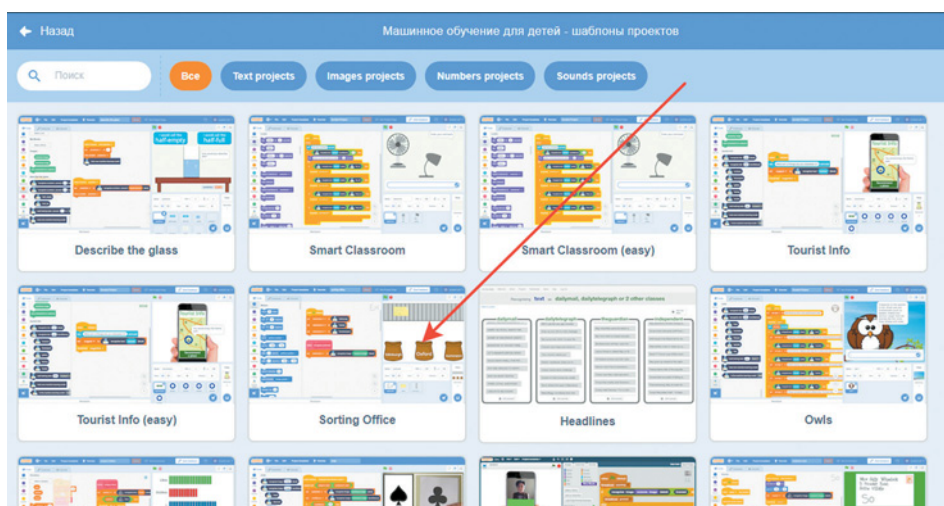


Рис. 6.13. Шаблон проекта «Sorting Office»

6. Нажмите на вкладку «Сцена», как показано на рисунке 6.14.

7. Теперь перейдите на вкладку «Фоны», как показано на рисунке 6.15.

8. Используйте инструмент «Текст», чтобы изменить названия на почтовых мешках. Переименуйте все три мешка, по названиям городов, которые вы выбрали для этого проекта. Если полное название города слишком длинное и не помещается на мешке, можете написать аббревиатуру этого города.

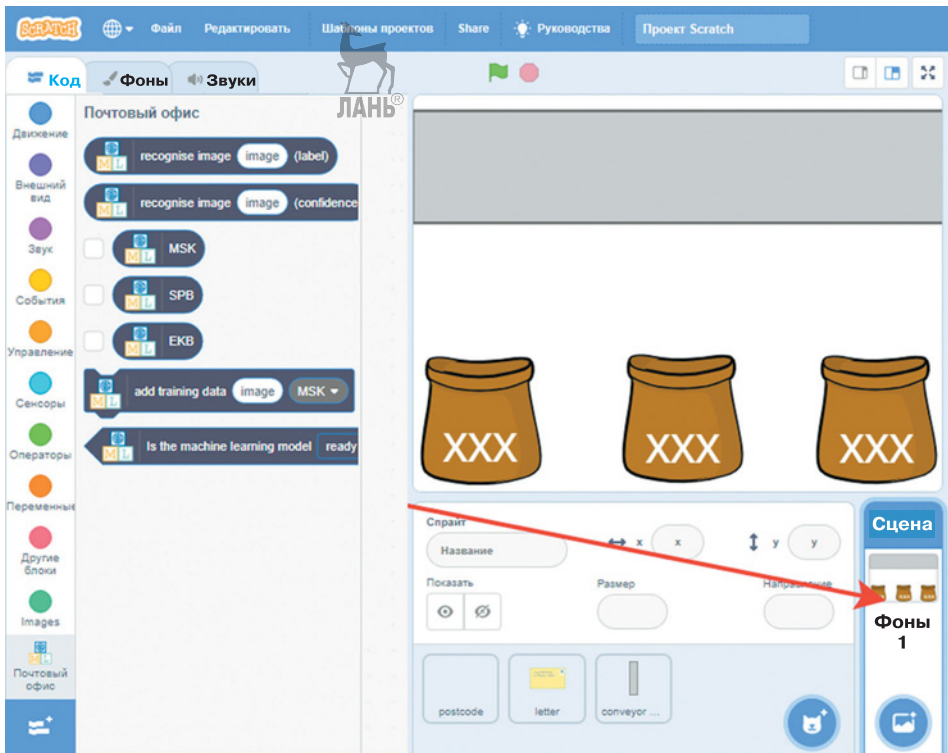


Рис. 6.14. Нажмите на вкладку «Сцена» в проекте «Sorting Office»

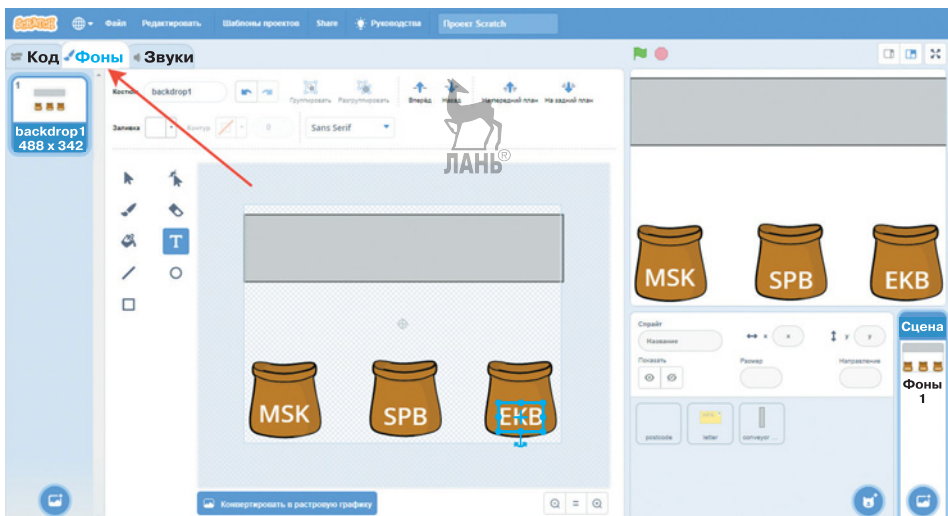


Рис. 6.15. Измените названия на почтовых мешках, чтобы они соответствовали названиям выбранных вами городов

9. Нажмите на спрайт «postcode» (в переводе с англ.: почтовый индекс, почтовый регион), а затем перейдите на вкладку «Костюмы», как показано на рисунке 6.16.

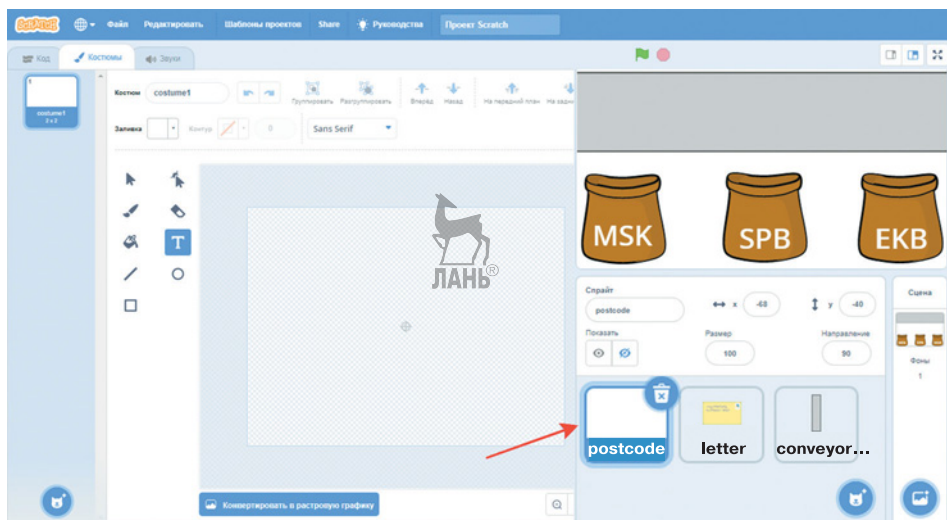


Рис. 6.16. Найдите спрайт «postcode» в списке спрайтов

10. Используйте инструмент «Кисть», чтобы изобразить на холсте надпись, соответствующую аббревиатуре одного из выбранных вами городов.

Результаты будут лучше, если будете использовать тот же стиль линии, что и в обучающих примерах. Поэтому выберите для параметра «Заливка» (Fill) черный цвет и задайте ширину линии (Stroke width) примерно 20, как показано на рисунке 6.17.

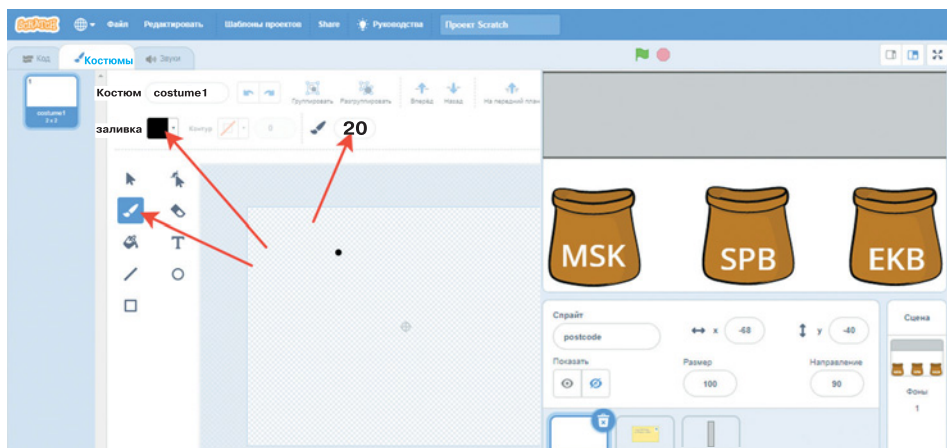


Рис. 6.17. Задайте параметры рисования, чтобы они соответствовали обучающим примерам

11. Когда закончите, нажмите на кнопку «Рисовать» в левом нижнем углу экрана, чтобы добавить новый костюм (рис. 6.18).

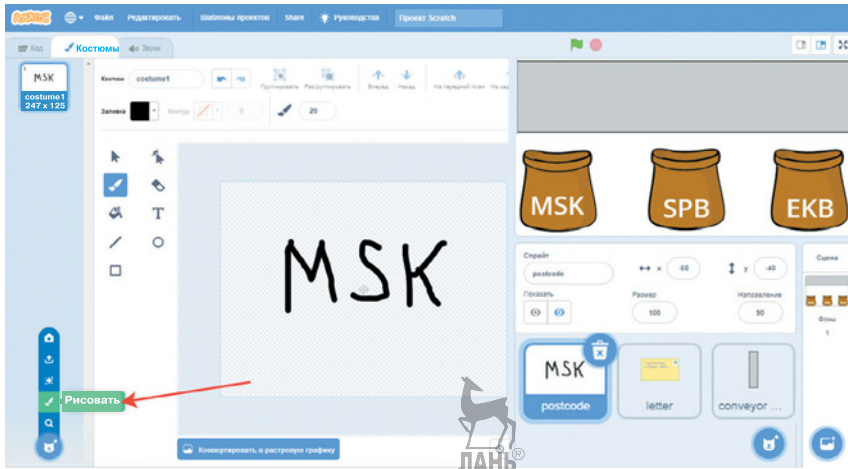


Рис. 6.18. Нажмите на кнопку «Рисовать», чтобы добавить новый костюм спрайту «postcode»



Примечание

Убедитесь, что вы нажимаете кнопку «Рисовать» для добавления нового костюма, а не нового спрайта.

12. Повторяйте шаги 10 и 11, пока не создадите несколько костюмов для спрайта «postcode». Нарисуйте аббревиатуру каждого из городов несколько раз, как показано на рисунке 6.19.

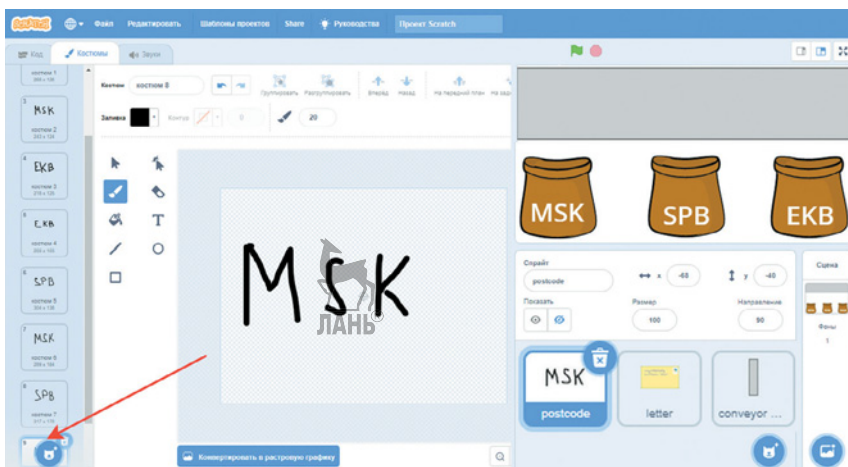


Рис. 6.19. Нарисуйте несколько тестовых костюмов для спрайта «postcode»

Не волнуйтесь, если не удастся сразу создать читаемую надпись, вы всегда можете нажать на голубую стрелку «Отменить» рядом с именем костюма и вернуться на несколько шагов назад.

13. Перейдите на вкладку «Код» и найдите фрагмент скрипта «Когда нажат зеленый флажок» (рис. 6.20).

Если сразу не получается найти нужный скрипт, прокрутите область скриптов — нужный вам фрагмент должен быть в левом верхнем углу.

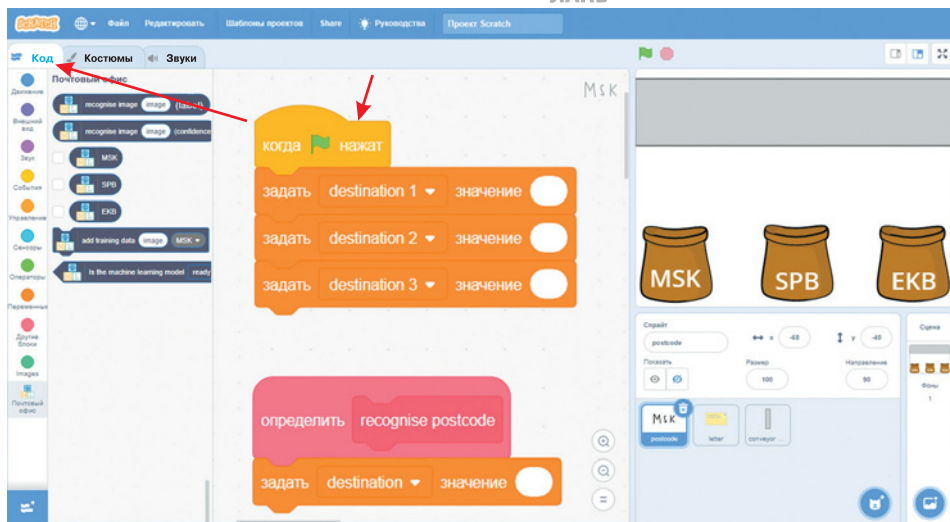


Рис. 6.20. Найдите фрагмент скрипта «Когда нажат зеленый флажок» для спрайта «postcode»



Примечание

Вы должны делать все это именно для спрайта «postcode». Если вы случайно нажали на другой спрайт, снова выберите «postcode» в списке спрайтов в нижней части экрана.

14. Перетащите в найденный фрагмент скрипта «когда нажат зеленый флажок» блоки с названиями выбранных вами городов, как показано на рисунке 6.21.

Обратите внимание — этот проект содержит несколько фрагментов скрипта «когда нажат зеленый флажок», поэтому найдите тот, который выглядит как на рисунке 6.21.

Также очень важно, чтобы вы расположили блоки с названиями городов в том же порядке, как вы написали их на почтовых мешках. Левый мешок должен соответствовать названию для destination 1, средний — destination 2, правый — destination 3.

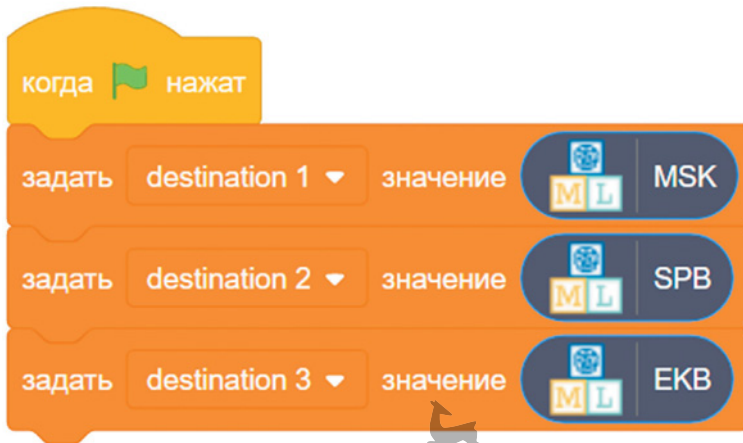


Рис. 6.21. Добавьте блоки с названиями выбранных вами городов в проект

15. Найдите фрагмент скрипта «recognise postcode» (в переводе с англ.: распознать почтовый индекс или регион) в области скриптов. Он должен располагаться рядом с фрагментом из предыдущего шага. Обратите внимание, вы все еще работаете со скриптом для спрайта «postcode».

16. Перетащите блок «recognise image» во фрагмент скрипта «recognise postcode», а затем перетащите в него блок «costume image», как показано на рисунке 6.22.

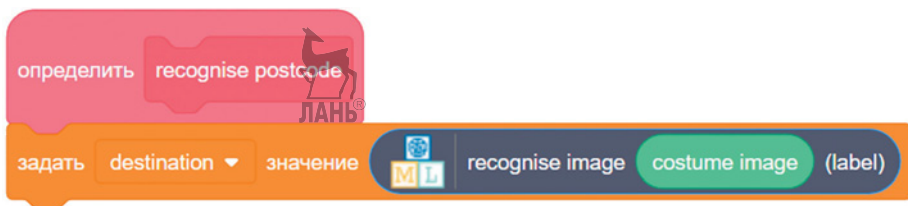


Рис. 6.22. Скрипт для распознавания аббревиатуры на конверте

Тестирование модели

Настало время попробовать сортировать письма!

Нажмите на зеленый флажок, чтобы увидеть свою модель машинного обучения в деле.

На экране появится движущаяся конвейерная лента, на которой вы увидите нарисованные вами почтовые названия, как показано на рисунке 6.23.

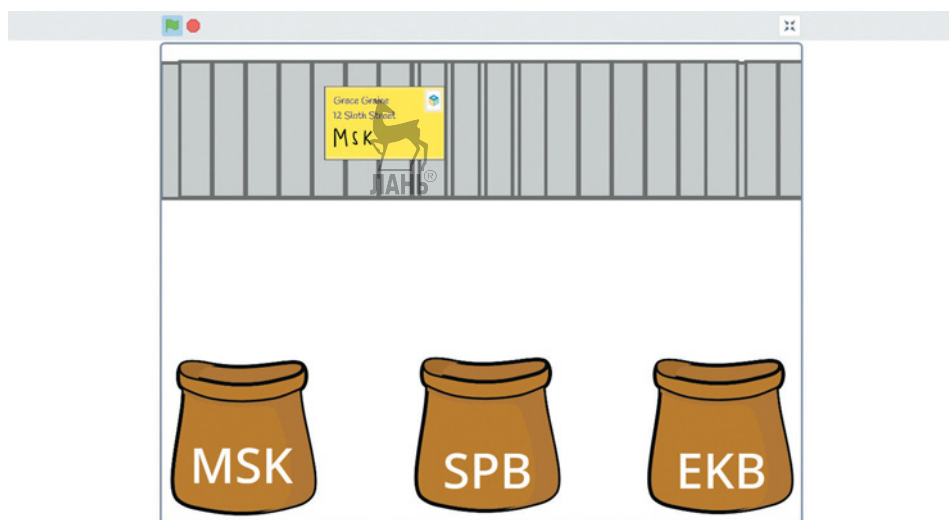


Рис. 6.23. Тестовый конверт на конвейерной ленте

Конверт будет увеличиваться, пока ваша модель машинного обучения пытается распознать то, что вы написали.

Как только изображение будет распознано, конверт отправится в соответствующий почтовый мешок, как показано на рисунке 6.24.

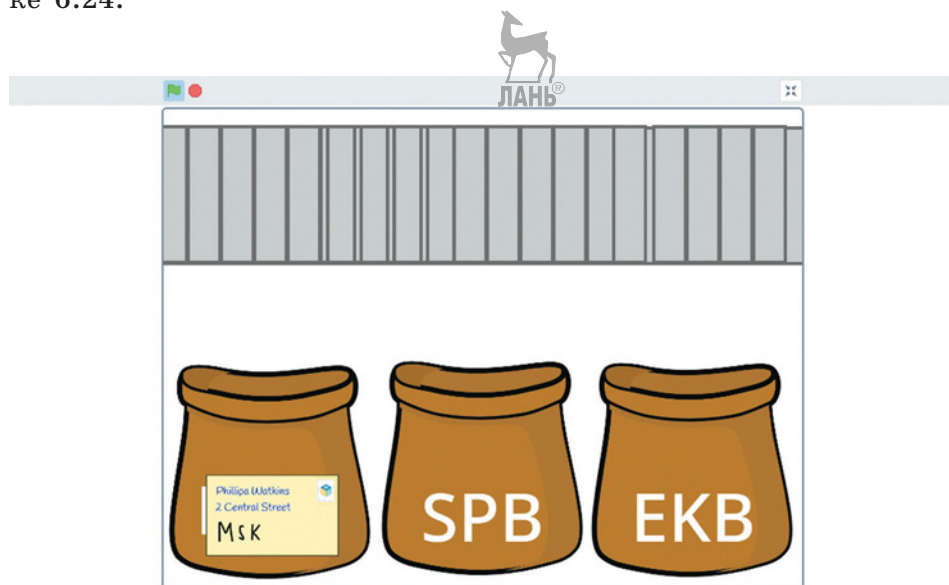


Рис. 6.24. Результат распознавания аббревиатур трех городов

Обзор и улучшение проекта

Вы обучили свою модель машинного обучения распознавать рукописный текст на конверте и создали проект Scratch, который использует технологию оптического распознавания символов (OCR) для автоматической сортировки писем.

Как можно улучшить этот проект?

Попросите кого-нибудь другого протестировать ваш проект. Сможет ли ваша модель распознать почерк других людей? Если результаты ухудшатся, вернитесь к этапу обучения модели и добавьте больше обучающих примеров в каждую коллекцию. (При этом не забудьте еще раз обучить обновленную модель.)

Чем более разнообразными будут обучающие примеры, тем лучше ваша модель будет распознавать разные стили написания и почерки.

Подумайте, что еще можно сделать, чтобы улучшить этот проект.

Что вы узнали

Сортировка почты является распространенным примером применения технологии оптического распознавания символов. Крупные сортировочные пункты по всему миру используют такие системы в своей работе для сортировки писем за доли секунды. В своем проекте вы реализовали распознавание аббревиатуры, но реальные системы оптического распознавания символов могут распознавать сразу несколько строк адреса. Основной принцип работы у них такой же. Подобные системы помогают сделать сортировку почты более эффективной и практичной в мировых масштабах.

Во всех предыдущих проектах этой книги вы обучали компьютер распознавать изображения. Но, как вы уже знаете, компьютер может научиться распознавать самые разные типы данных. В следующей главе вы научите его распознавать текст!





ГЛАВА 7

Распознавание эмоций в тексте

В этой главе вы узнаете, как обучить компьютер распознавать эмоции в тексте, и познакомитесь с технологией **анализа тональности текста** (sentiment analysis).

Представьте, что вы узнали, что завтра пойдете в зоопарк, и вас попросили написать несколько предложений о предстоящем событии.

Подумайте, что бы вы написали о предстоящем походе, если бы вас обрадовала эта новость. Вы же очень любите зоопарки и посещаете их в разных городах. Какие слова вы бы использовали? Повлияли бы ваши эмоции, ваша радость, ваше волнение на текст анонса?

А теперь представьте, что вы вовсе не рады, а, наоборот, недовольны или даже злитесь по этому поводу. Например, вы терпеть не можете зоопарки или же у вас были другие планы на завтра, и вас раздражает, что кто-то заставляет вас туда идти. Как это отразится на вашем тексте? Как вы думаете, заставит ли вас ваше раздражение подбирать другие слова или даже как-то по-другому формулировать предложения?

Смысл обоих текстов оставался бы тем же самым (вы завтра идете в зоопарк), но их содержание, тон и настроение были бы совершенно разными!

Оказывается, компьютер можно научить распознавать особенности текстов, написанных в разных эмоциональных состояниях — раздражении или, наоборот, в хорошем настроении. Если подготовить достаточно большое количество обучающих примеров текста, содержащего разные эмоции и чувства, можно обучить модель машинного обучения распознавать эмоциональную окраску текста на основе имеющихся лексических и грамматических шаблонов.

Системы машинного обучения, способные распознавать эмоции и тональность текста, используются для того, чтобы определить, как люди относятся к тем или иным вещам. Например, компании используют анализ тональности текста, чтобы узнать, что клиенты думают об их продуктах или услугах. Модели машинного обучения анализируют миллионы блогов, форумов, новостных групп и сообщ-

щений в социальных сетях — гораздо больше, чем эти компании могли бы проанализировать сами. Результаты такого анализа показывают, какую часть отзывов модель машинного обучения считает положительными, а какую — отрицательными, а также отбирают критические замечания и жалобы, встречающиеся чаще всего.

Кстати, анализ тональности текста можно применять не только к сообщениям во всем Интернете, но и внутри одной компании. Например, во многих государственных учреждениях это используется для сортировки электронной почты и корреспонденции, поступающей в службу поддержки пользователей, причем первыми обрабатываются письма и сообщения, которые система распознала как наиболее злые и раздраженные. (Не используйте это как совет к действию!)

Точно таким же образом компании используют анализ тональности текста и для внутренних чатов, чтобы оценить, насколько довольны их сотрудники и есть ли какие-то проблемы, требующие срочного вмешательства.

Давайте попробуем научить модель машинного обучения распознавать в тексте комплименты и оскорбления.

Выполнение проекта

В этом проекте вы создадите персонажа, который будет реагировать на ваши сообщения (рис. 7.1). Персонаж будет выглядеть радостным, если вы сделаете ему комплимент, и грустным, если вы оскорбите его.

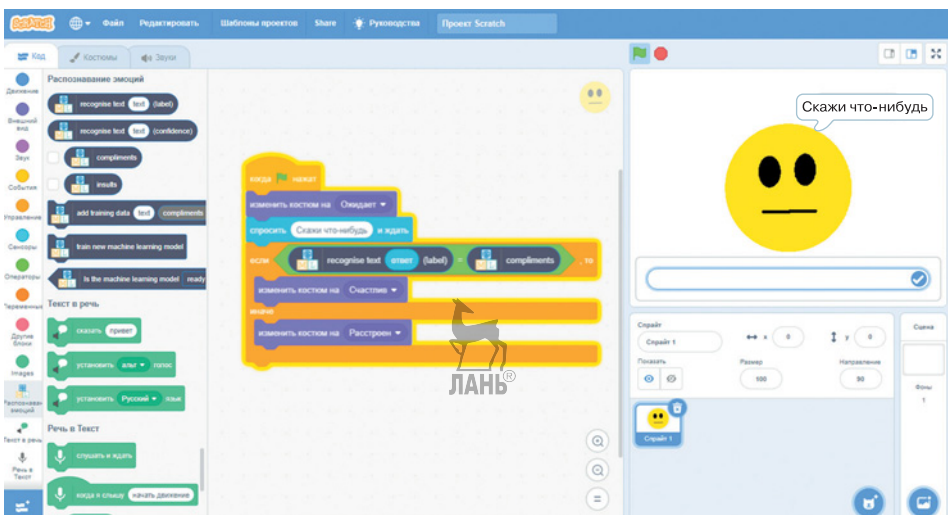


Рис. 7.1. Распознавание комплиментов и оскорблений

Подготовка проекта-игры

Первое, что вам нужно сделать, — это создать свой персонаж. Для своих примеров я нарисовал простое лицо. Вы можете нарисовать любой персонаж, но по его лицу должно быть понятно, радуется он или грустит. Это может быть животное, робот, инопланетянин или что-то другое.

1. Перейдите на страницу <https://machinelearningforkids.co.uk/scratch3/>, чтобы создать новый проект Scratch.

2. Перейдите на вкладку «Костюмы», как показано на рисунке 7.2.

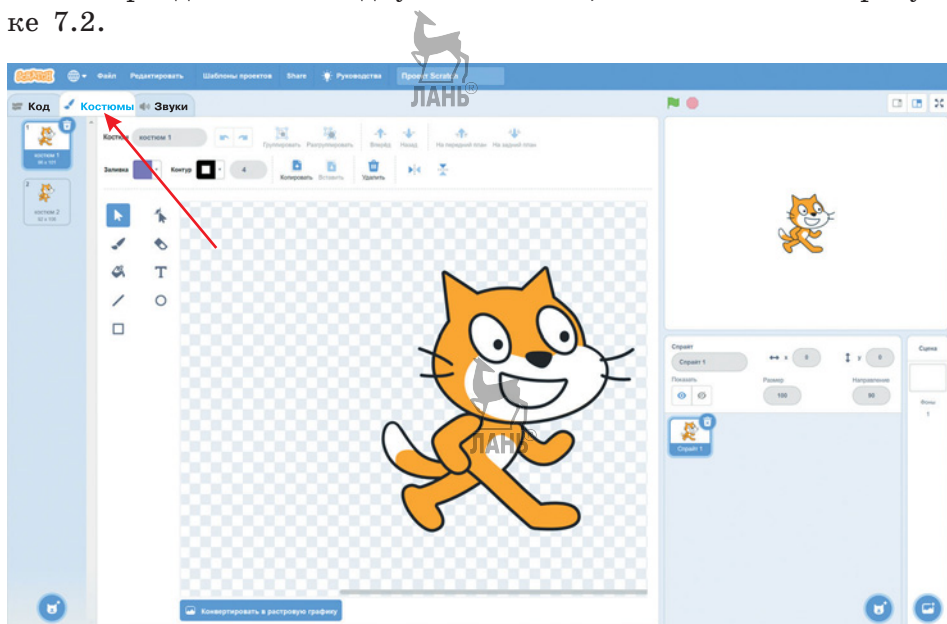


Рис. 7.2. Перейдите на вкладку «Костюмы» и создайте свой персонаж

3. Наведите указатель мыши на иконку с изображением кота в левом нижнем углу экрана, чтобы посмотреть все доступные способы создания костюма (рис. 7.3).

Если вы хотите нарисовать свой собственный персонаж, нажмите на кнопку «Рисовать». Для своего проекта из нескольких кругов и овалов я создал зеленого инопланетянина и также нарисовал ему прическу с помощью кисти (рис. 7.4).

Если же вы не хотите рисовать, есть еще несколько вариантов. Например, вы можете нажать на кнопку «Камера» и сделать фотографию с помощью веб-камеры. Также вы можете нажать на кнопку «Загрузить костюм» и использовать любое изображение,

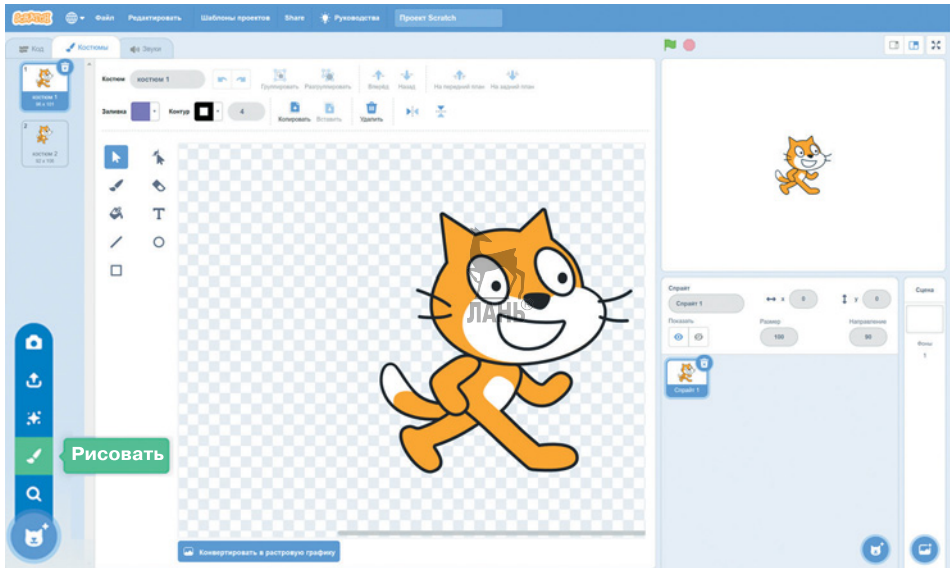


Рис. 7.3. Добавление нового костюма

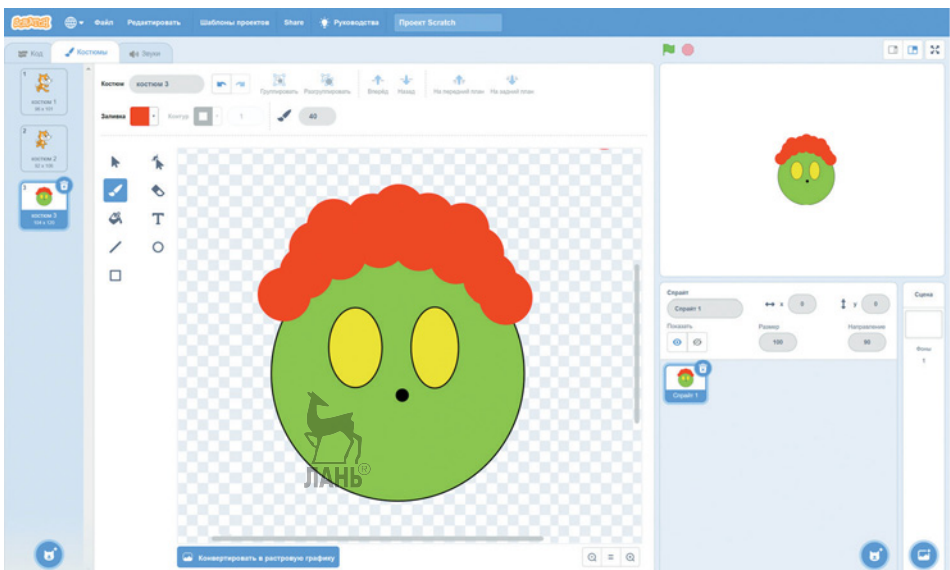


Рис. 7.4. Вы можете нарисовать собственный персонаж

сохраненное на вашем компьютере. Или же вы можете нажать на кнопку «Выбрать костюм» и использовать один из множества костюмов, доступных в библиотеке Scratch.

Независимо от того, какой из вариантов вы выберете, после окончания работы над созданием своего персонажа он должен появиться на холсте.

4. Найдите костюм, который вы только что создали, в панели костюмов, щелкните по нему правой кнопкой мыши и выберите команду «Дублировать», как показано на рисунке 7.5. Вам понадобятся три копии этого костюма.

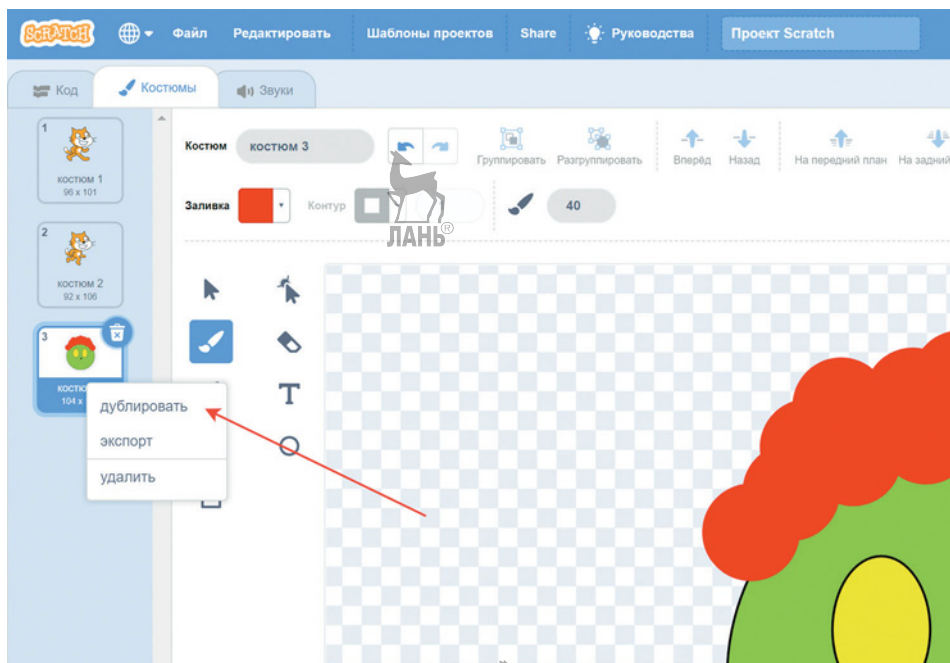


Рис. 7.5. Дублирование костюма



Примечание

Убедитесь, что вы дублируете именно костюмы, а не спрайты. Вам нужен один спрайт с тремя костюмами, а не три разных спрайта.

5. Переименуйте каждую из созданных копий костюма. Для этого нажмите на каждую из них в панели костюмов, а затем измените их имена в текстовом поле «Костюм», как показано на рисунке 7.6. Назовите их «ожидает», «счастлив» и «расстроен».

6. Теперь нужно сделать так, чтобы костюмы соответствовали своим названиям. Для этого нажмите последовательно на каждый из костюмов и дорисуйте его, добавив ему нужное выражение лица, как показано на рисунке 7.7.

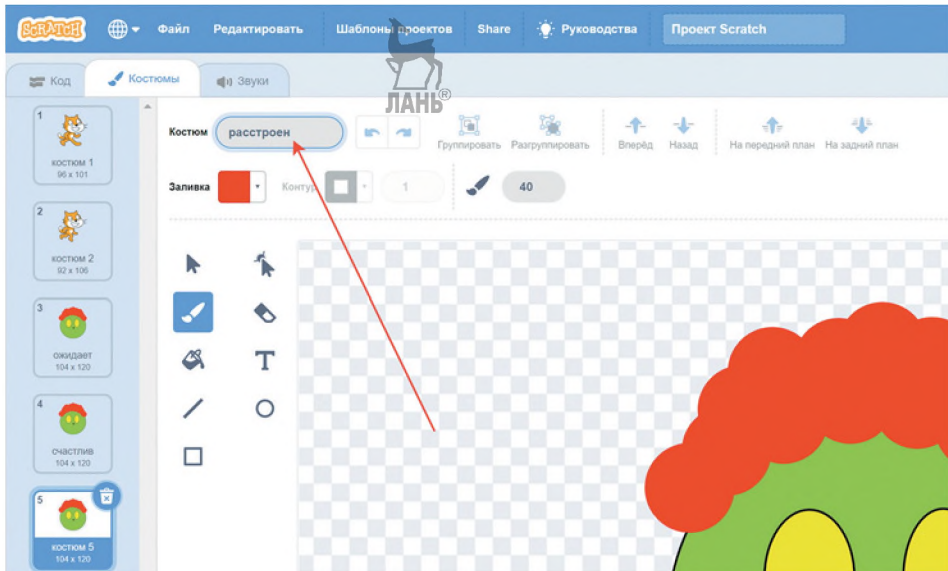


Рис. 7.6. Переименование костюма

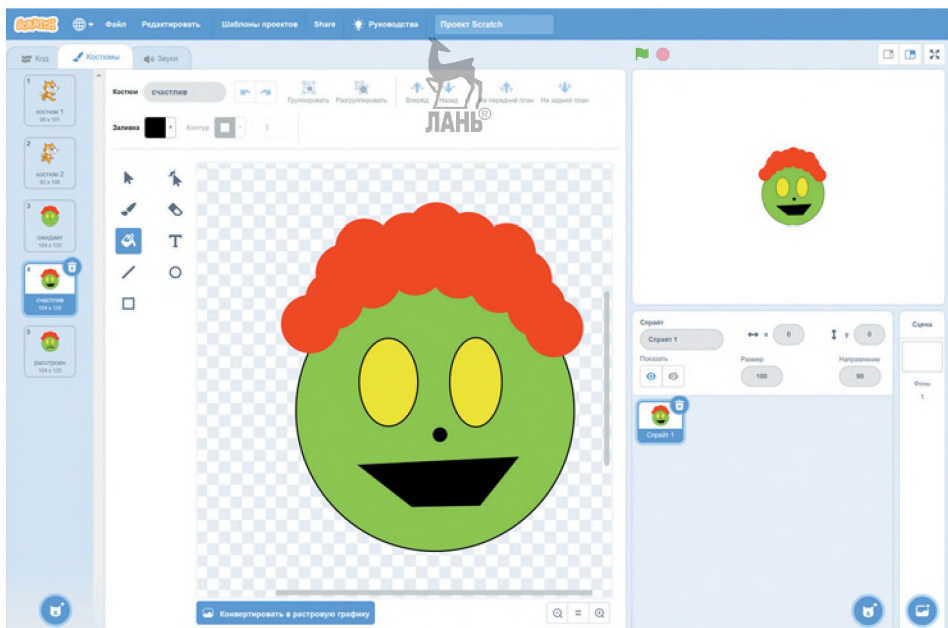


Рис. 7.7. Добавление разных выражений лица костюмам персонажа

Костюм «счастлив» должен выглядеть радостным. Если это лицо, вы можете нарисовать на нем улыбку. Если это животное, можно изменить положение его хвоста или ушей. Или же вы

можете просто дорисовать своему персонажу табличку, на которой написано, что он счастлив, либо добавить смайлик.

Костюм «расстроен» должен выглядеть грустным. Если это лицо, вы можете нарисовать на нем нахмуренные брови или слезу.

Костюм «ожидает» будет отображаться на экране, пока персонаж будет ждать вашего сообщения. Поэтому он не должен выглядеть ни счастливым, ни расстроенным.

7. Сохраните проект Scratch, выбрав в командном меню **Файл → Сохранить на свой компьютер!**

Программирование игры без использования машинного обучения

Вам будет проще понять, насколько быструю и эффективную работу выполняет машинное обучение, если сначала вы попытаетесь самостоятельно написать программу для этого проекта без его использования. Но вы можете пропустить эту часть и сразу перейти к созданию и использованию модели машинного обучения. Хотя так будет менее интересно.

1. Перейдите на вкладку «Код», как показано на рисунке 7.8.
2. Создайте такой же скрипт, как показан на рисунке 7.9.



Примечание

Если вы не знаете, как создаются скрипты в Scratch, возвратитесь к разделу «Программирование в среде Scratch» на с. 15.

3. Сохраните проект, выбрав в командном меню **Файл → Сохранить на свой компьютер.**

4. Запустите и протестируйте свой проект. Нажмите на зеленый флажок. На экране появится персонаж и попросит вас написать что-нибудь для него. Если вы введете фразы «Ты мне нравишься» или «Ты красивый», персонаж будет выглядеть счастливым. Если же вы напечатаете что-либо другое — он будет выглядеть расстроенным.

Теперь напишите фразу «Ты милый». Она приятная, но почему же персонаж выглядит расстроенным?

Чтобы это изменить, вам придется изменить скрипт и добавить в него третье условие. Тогда персонаж будет выглядеть счастливым, если вы напишете ему одну из фраз: «Ты мне нравишься», «Ты красивый» или «Ты милый».

Как вы думаете, сможете ли вы изменить скрипт таким образом, чтобы в условии содержались все возможные формулировки комплиментов и оскорблений?

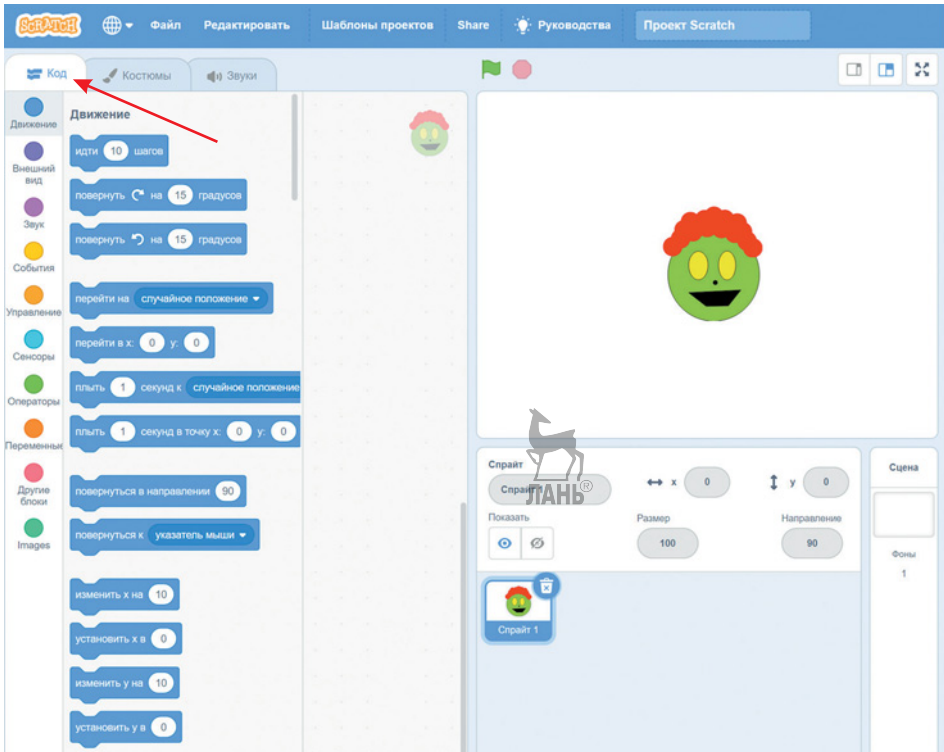


Рис. 7.8. Перейдите на вкладку «Код»



Рис. 7.9. Программирование игры без использования машинного обучения

Как мы обсуждали в главе 1, *машинное обучение* не единственный способ создания систем искусственного интеллекта. Вы только что создали программу, используя подход *написания четких пошаговых инструкций*. И теперь вы можете понять, почему, хотя такой подход до сих пор используется при создании некоторых систем, машинное обучение является более предпочтительным способом реализации сложных проектов. Но это не искусственный интеллект! Теперь вы создадите и обучите модель машинного обучения для реализации этого же проекта, а в конце главы сравните эти два подхода.

Обучение модели

Для того чтобы обучить компьютер распознавать комплименты и оскорбления, вам нужно собрать обучающие примеры текстов, содержащих и то и другое, а затем использовать их для обучения модели машинного обучения.

1. Создайте новый проект машинного обучения* и назовите его «Распознавание эмоций». В качестве распознаваемого типа данных выберите «текст», после чего выберите язык, на котором будет написан распознаваемый текст, — «Another language» (в переводе с англ.: другой язык).



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 7.10.

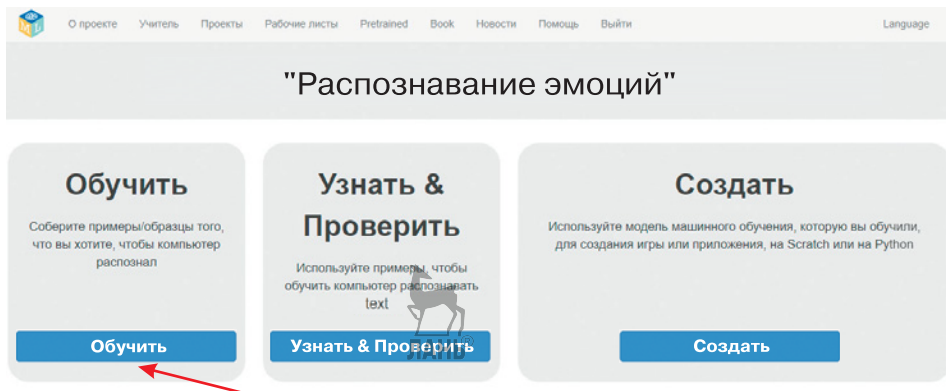


Рис. 7.10. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

* Если вы находитесь в РФ, то используйте пробный режим.

3. Нажмите на кнопку «Добавить новую метку» (рис. 7.11) для создания первой коллекции и назовите ее «compliments» (в переводе с англ.: комплименты). Затем создайте еще одну коллекцию с именем «insults» (англ.: оскорбления).

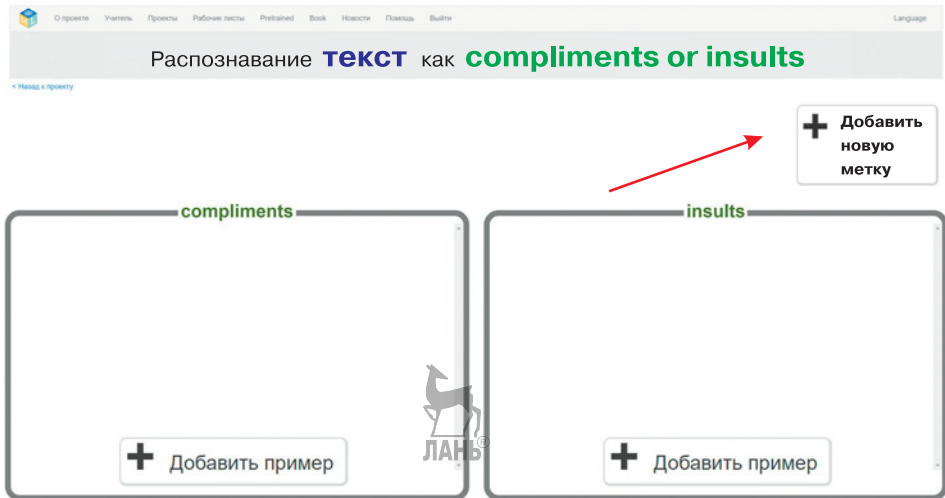


Рис. 7.11. Нажмите на кнопку «Добавить новую метку» для создания новой коллекции примеров

4. Нажмите на кнопку «Добавить пример», чтобы добавить новый обучающий пример в коллекцию комплиментов (рис. 7.12). Напишите любой комплимент, который придет вам в голову.

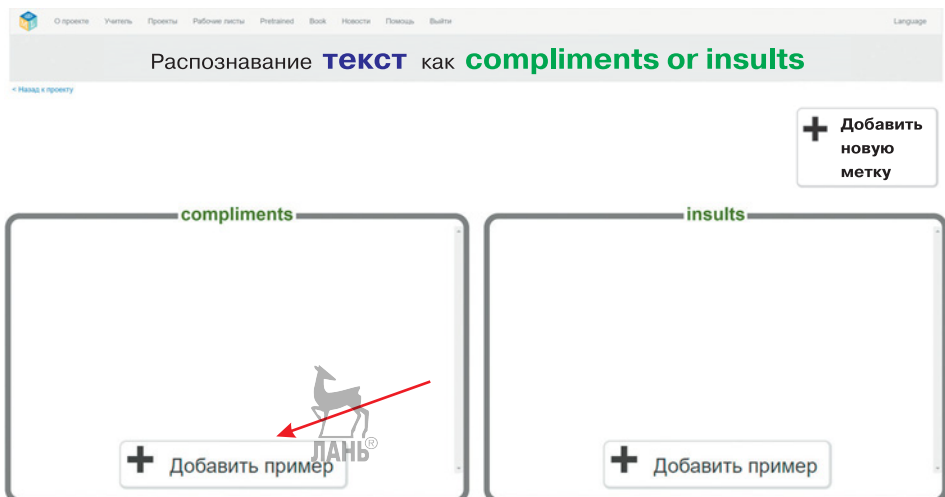


Рис. 7.12. Обучающие примеры для распознавания комплиментов

Повторяйте этот шаг до тех пор, пока в коллекции не появится хотя бы пять примеров комплиментов, которые смогут порадовать ваш персонаж. Эти примеры ваша модель будет использовать для обучения, поэтому постарайтесь сделать их как можно более разнообразными.

5. Нажмите на кнопку «Добавить пример», чтобы добавить новый обучающий пример в коллекцию оскорблений (рис. 7.13). Напишите любую неприятную, оскорбительную фразу, которая может расстроить ваш персонаж. Но не ругайтесь слишком сильно!

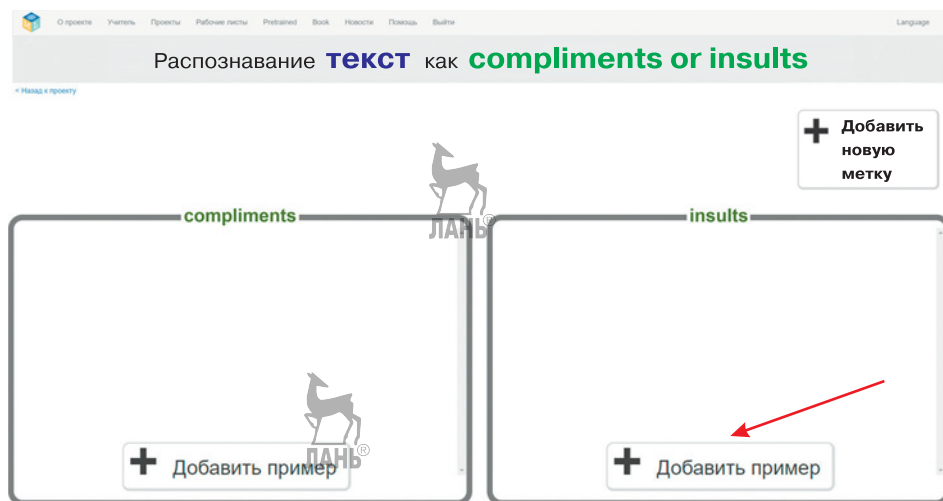


Рис. 7.13. Обучающие примеры для распознавания оскорблений

Повторяйте этот шаг до тех пор, пока в коллекции не наберется хотя бы пять обучающих примеров. И снова позаботьтесь об их разнообразии, чтобы модель лучше распознавала тестовые примеры.

6. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

7. Нажмите на кнопку «Узнать и проверить», чтобы перейти к следующему этапу проекта.

8. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 7.14).

Компьютер будет использовать собранные вами обучающие примеры, чтобы научиться распознавать комплименты и оскорбления.

Для этого он будет искать общие закономерности в примерах из каждой коллекции. Причем искать эти закономерности он будет не только в словах, но и в способах формулировки предложений. Затем он будет пытаться найти эти же закономерности в

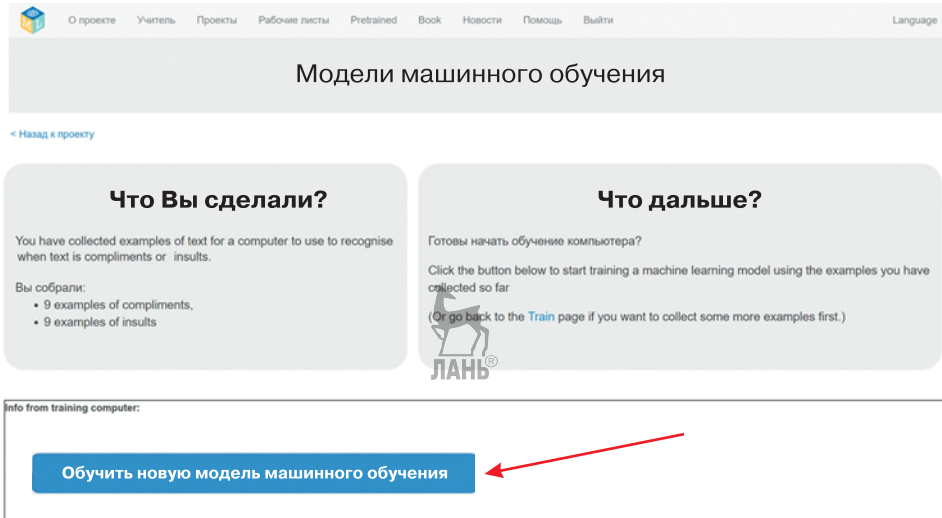


Рис. 7.14. Нажмите на кнопку «Обучить новую модель машинного обучения», чтобы начать обучение

предложенных тестовых примерах для распознавания смысла сообщений, которые вы будете отправлять ему в последующих шагах.

Обучение модели может занять около минуты, но вы заметите, что оно пройдет гораздо быстрее, чем обучение модели для распознавания изображений. Действительно, компьютеру гораздо легче научиться распознавать закономерности в тексте, чем на изображениях.

9. Протестируйте свою модель. Для этого наберите фразу-комплимент или фразу-оскорбление в поле для ввода текста, как показано на рисунке 7.15.

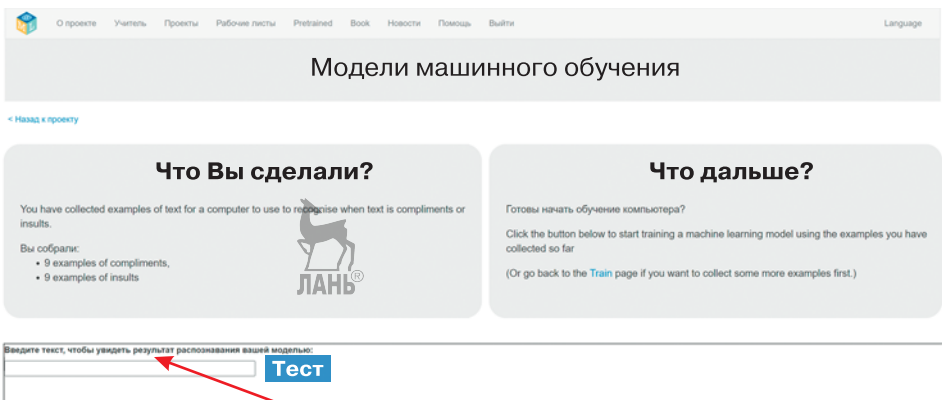


Рис. 7.15. Тестирование — очень важная часть машинного обучения

Очень важно, чтобы тестовые примеры не совпадали с обучающими примерами. Вам нужно понять, насколько хорошо ваша модель обучилась распознавать новые примеры, которые ей еще не попадались, а не насколько хорошо она научилась запоминать обучающие примеры.

Если модель допускает много ошибок, вернитесь к этапу обучения и пополните каждую коллекцию обучающих примеров разнообразными фразами. И снова обучите модель на новых примерах.

Делайте это до тех пор, пока не останетесь довольны результатами работы модели. В следующей главе вы узнаете о более эффективных способах тестирования модели, но пока достаточно ввести несколько разных тестовых примеров.

Программирование игры с использованием машинного обучения

Теперь, когда у вас есть своя модель машинного обучения, способная распознавать комплименты и оскорбления, вы можете изменить созданный ранее проект, добавив в него машинное обучение вместо пошаговых инструкций, которые вы написали ранее.

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.
2. Нажмите на кнопку «Создать» (рис. 7.16).

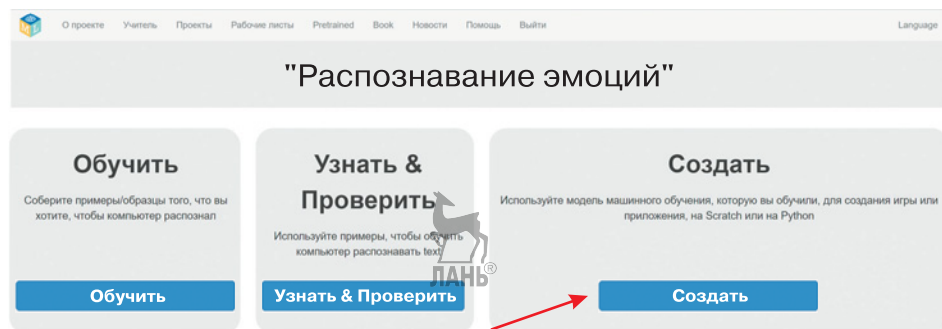


Рис. 7.16. Если модель хорошо справляется с распознаванием тестовых примеров, настало время использовать ее в проекте!

3. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открыть новое окно со средой Scratch.
4. На Панели инструментов вы увидите новую категорию блоков, которая будет называться так же, как и ваш проект (рис. 7.17).

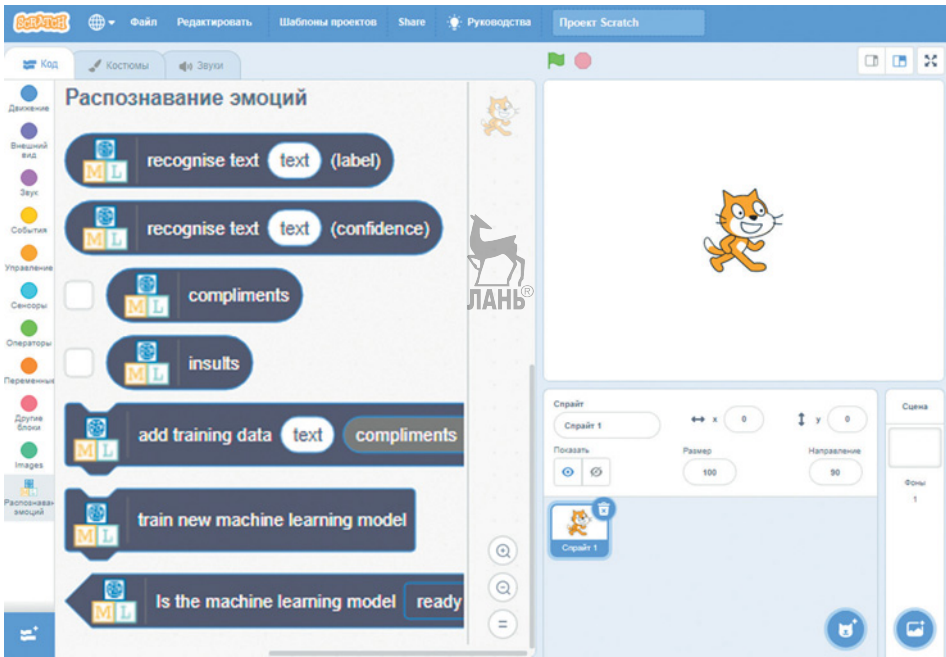


Рис. 7.17. Новые блоки для вашего проекта машинного обучения автоматически добавятся на Панель инструментов Scratch

5. Откройте проект, который вы скачали на свой компьютер ранее. Для этого выберите в Главном меню **Файл** → **Загрузить с компьютера**, как показано на рисунке 7.18.

6. Если вы создавали проект без использования машинного обучения, доработайте скрипт таким образом, чтобы он выглядел как на рисунке 7.19. Если вы пропустили этот этап, создайте этот скрипт.



Примечание

Если вы не знаете, как создаются скрипты в Scratch, возвращитесь к разделу «Программирование в среде Scratch» на с. 15.

В этом скрипте ваш персонаж попросит вас написать для него сообщение. Скрипт использует вашу модель машинного обучения, чтобы распознать введенное вами сообщение и определить, содержит оно комплименты или оскорбления. Затем он сменит костюм на тот, который будет соответствовать распознанной эмоции, создавая впечатление, что персонаж реагирует на ваши сообщения.

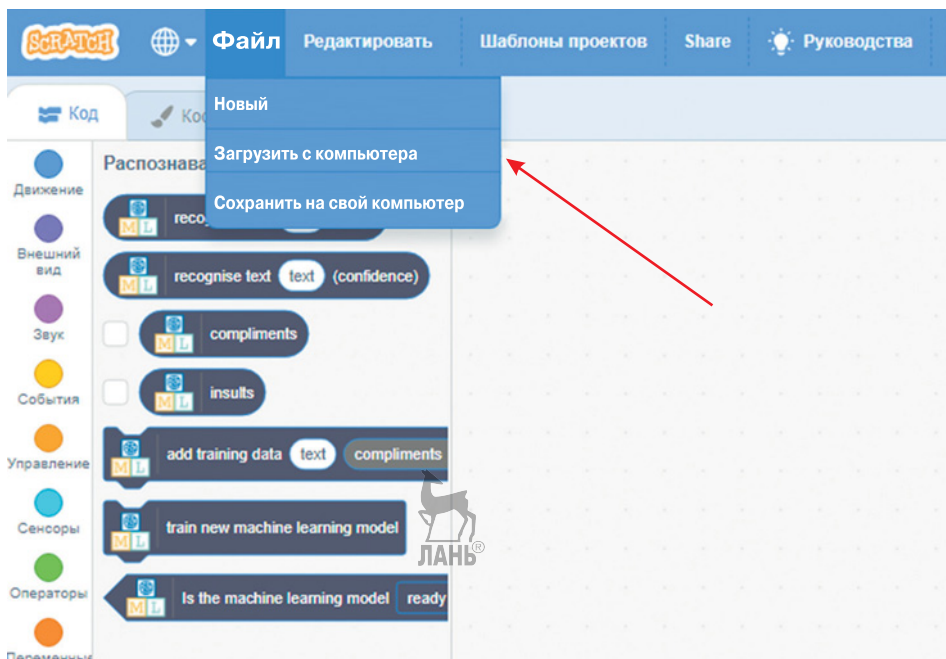


Рис. 7.18. Откройте проект, который вы скачали на свой компьютер ранее

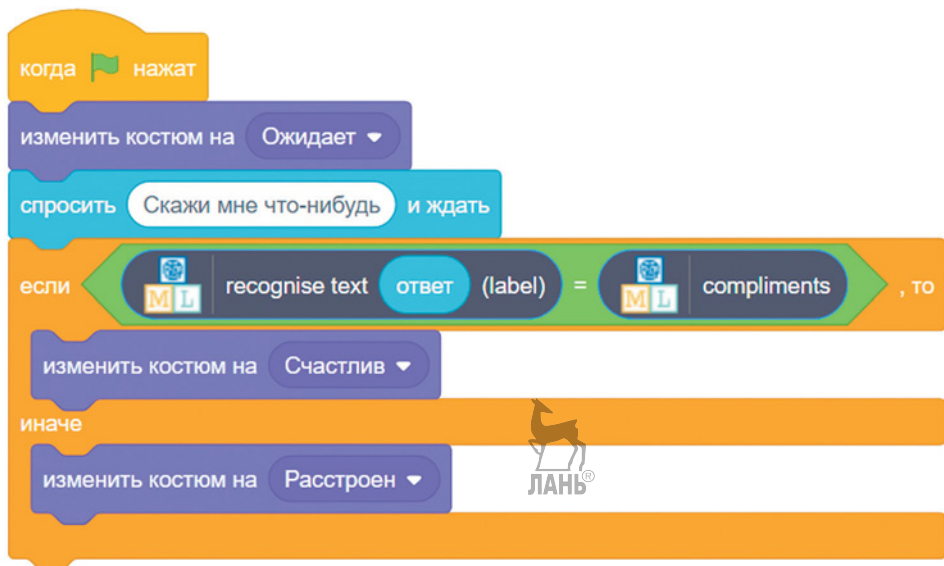


Рис. 7.19. Программирование игры с использованием машинного обучения

Если вы создавали этот же проект без использования машинного обучения, сравните этот скрипт с предыдущим. Как вы можете убедиться, машинное обучение существенно упростило создание игры, которая может распознавать более широкий спектр возможных сообщений.

Тестирование игры

Настало время протестировать ваш проект-игру! Нажмите на зеленый флажок и попробуйте набрать несколько сообщений для персонажа. Даже если вы придумаете фразы, которые не использовали в качестве обучающих примеров, персонаж должен реагировать на них корректно. Если этого не происходит, вы всегда можете вернуться к этапу обучения и добавить больше обучающих примеров, а затем обучить модель снова.

Итак, вы успешно создали персонаж, который научился распознавать и реагировать на комплименты и оскорбления, посылаемые вами.



Обзор и улучшение проекта

Давайте рассмотрим несколько способов, как можно улучшить этот проект.



Использование голосовых сообщений вместо напечатанных текстов

А как насчет того, чтобы усовершенствовать этот проект и научить его распознавать комплименты и оскорбления, которые вы произносите вслух?

Для этого вам понадобится микрофон. Кроме того, нужно будет подключить к проекту расширение «Speech to Text» (в переводе с англ.: преобразование речи в текст) из библиотеки расширений Scratch. Чтобы получить доступ к библиотеке расширений, нажмите на кнопку «Добавить расширение» в нижней части Панели инструментов (кнопка выглядит как два блока и знак «+» между ними). Эта библиотека содержит дополнительные блоки, которые вы можете использовать в своих проектах.

Найдите нужное расширение по названию и нажмите на него, чтобы добавить содержащиеся в нем блоки на вашу Панель инструментов. Затем обновите скрипт, чтобы он выглядел как на рисунке 7.20.

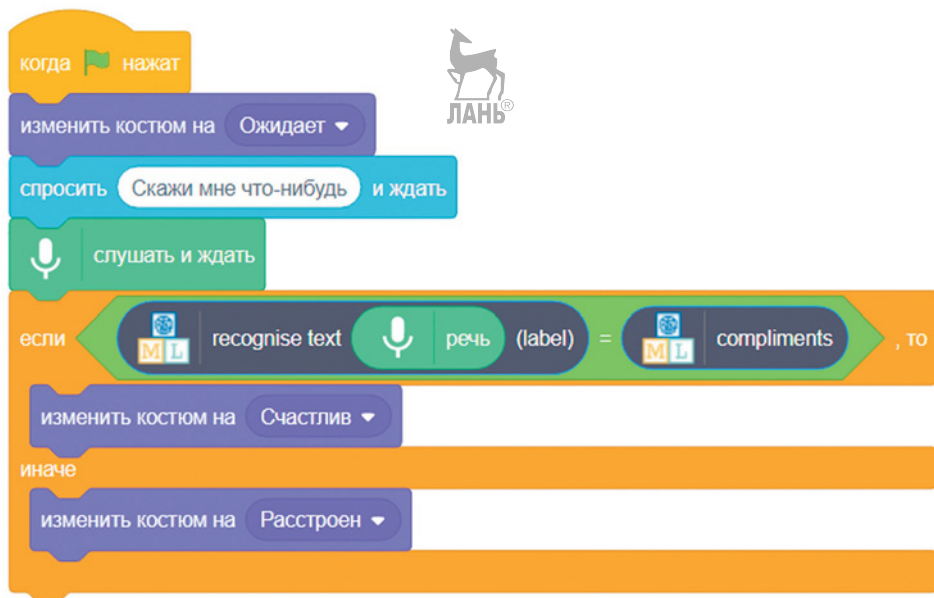


Рис. 7.20. Обновленный проект, использующий распознавание голоса



Примечание

На момент написания книги расширение «Speech to Text» можно было добавить только при работе в веб-браузере Google Chrome.

Распознавание речи — это еще один популярный способ использования машинного обучения. В этот раз вы не обучали модель распознавания речи самостоятельно, а только использовали модель, которую кто-то обучил за вас. Но в целом основной принцип создания речевых блоков у нее почти такой же, как и у создания печатных примеров, которые создавали вы.

А как вы думаете, можно ли улучшить этот проект?

Распознавание речи, которая не содержит ни оскорблений, ни комплиментов

Напишите своему персонажу сообщение «Который час?». Он может подумать, что это комплимент, и обрадоваться. Или же, наоборот, он может подумать, что это оскорбление, и расстроиться.

Но на самом деле ни одна из этих реакций не будет правильной. Поэтому вы можете улучшить свой проект и сделать так, чтобы персонаж никак не реагировал, если полученное им сообщение не содержит ни оскорблений, ни комплиментов.

Когда вы тестировали свою модель на этапе «Узнать и проверить», вы наверняка обратили внимание на **степень уверенности (score confident)**, которую компьютер отображает, чтобы показать, насколько он уверен в полученном результате распознавания тестового примера.

Попробуйте снова ввести тестовый пример «Который час?», как показано на рисунке 7.21.

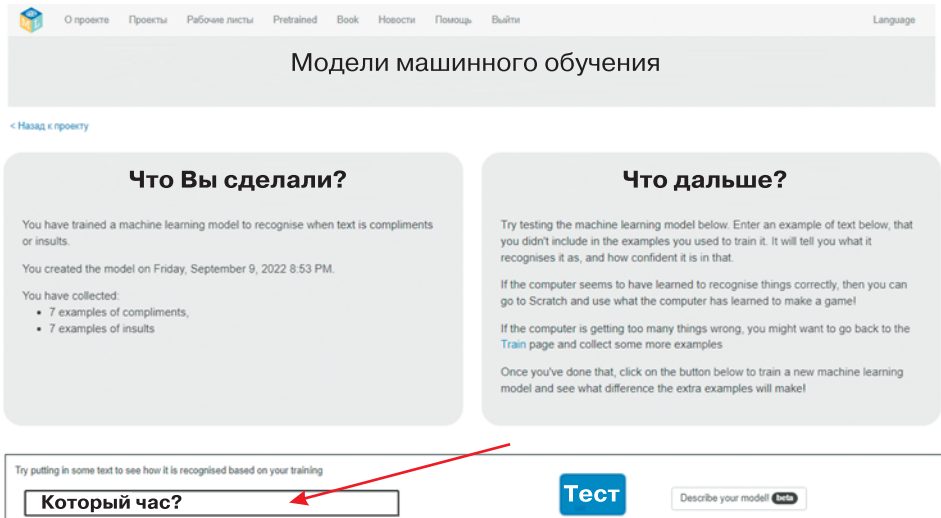


Рис. 7.21. Проверьте, насколько уверен будет компьютер в своем варианте при распознавании сообщения «Который час?»

Вы увидите, что для этого сообщения отобразится очень низкий процент уверенности. Таким способом модель пытается показать вам, что она не распознала предложенный текст. Это означает, что в обучающих примерах модель не нашла ничего подобного, поэтому она не смогла точно распознать, комплимент это или оскорбление.

Моя модель показала степень уверенности 0%, но ваш результат может быть выше, в зависимости от того, как вы обучали свою модель. Например, если среди ваших обучающих примеров было что-то вроде «Что с тобой не так?», при тестировании примера «Который час?» ваша модель может отнести эту фразу к оскорблениям, поскольку это тоже вопрос, со степенью уверенности 10%. Однако эта информация также может быть полезной, поскольку модель сообщила вам, что на 90% это не оскорбление. Это значит, что тестовое сообщение имеет некоторые общие закономерности с оскорблениями, но не может быть с уверенностью распознано как одно из них.

Поэкспериментируйте с различными тестовыми примерами, которые не содержат ни оскорблений, ни комплиментов. Обращайте внимание на степень уверенности, которую будет показывать ваша модель. Сравните эти значения со значениями степени уверенности, которые показывает ваша модель, если тестовые примеры действительно содержат оскорбления или комплименты.

Какую степень уверенности показывает ваша модель машинного обучения, когда она правильно распознает фактический комплимент или оскорбление?

Вы также можете добавить отображение степени уверенности в свой проект Scratch. Измените скрипт, как показано на рисунке 7.22.

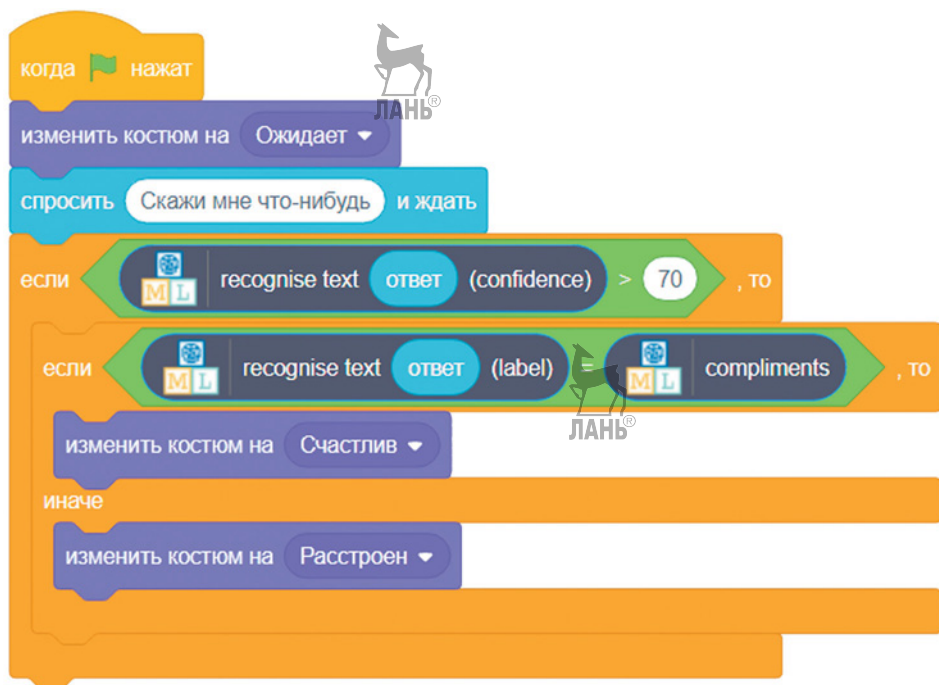


Рис. 7.22. Добавьте отображение показателя степени уверенности в свой проект

Выполняя этот скрипт, персонаж будет реагировать на сообщение только в том случае, если степень уверенности при его распознавании будет выше 70. В остальных случаях персонаж проигнорирует сообщение.

Вы можете также изменить значение 70 на то значение, которое чаще всего показывала ваша модель при тестировании.

Может быть, можно еще улучшить этот проект?

Обучение на ошибках



Когда вы используете системы машинного обучения, вы чаще всего понимаете, допустил ли компьютер ошибку. Один из способов улучшить ваш проект — это научить его «обучаться на ошибках».

Можно изменить проект так, чтобы пользователь мог выражать свое согласие или несогласие с результатами работы модели. Например, это могут быть кнопки «Согласен» и «Не согласен» или текстовое поле, в которое нужно будет ввести ответ «да» или «нет».

Скрипт, который показан на рисунке 7.23, спрашивает пользователя, права ли модель машинного обучения. Если пользователь напечатает «нет», то сообщение, которое было распознано некорректно, будет добавлено в коллекцию обучающих примеров. После добавления каждого пяти новых примеров модель будет обучаться заново.

Научив свою модель обучаться на ошибках, вы делаете ее все более и более эффективной с каждой минутой использования.

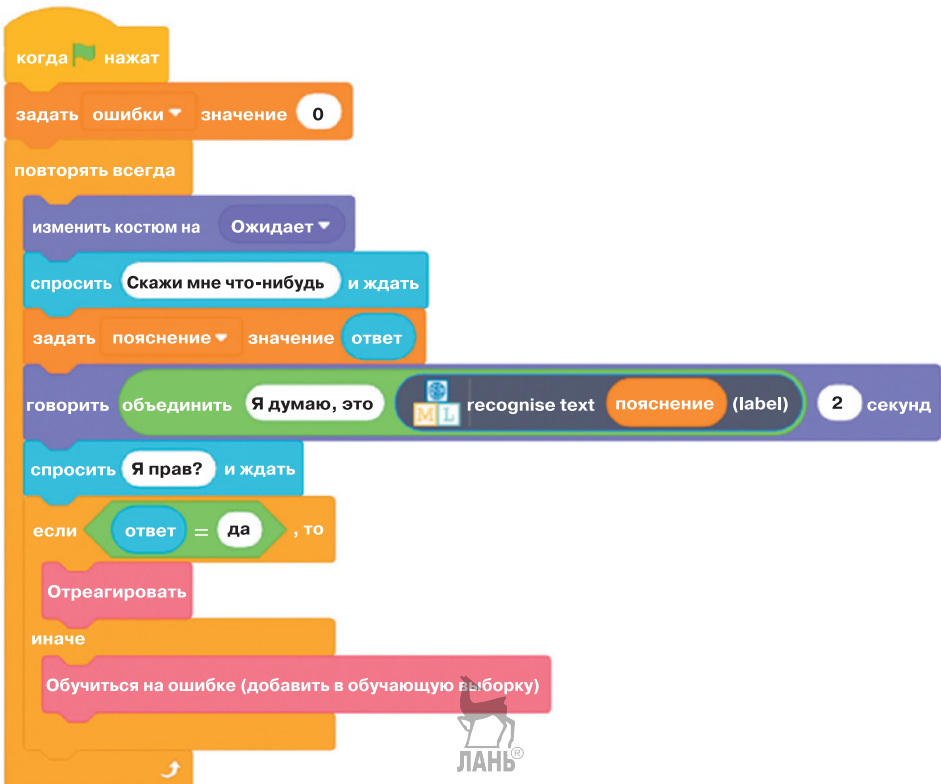


Рис. 7.23. Пример скрипта для обучения модели на ошибках
(окончание на с. 124)

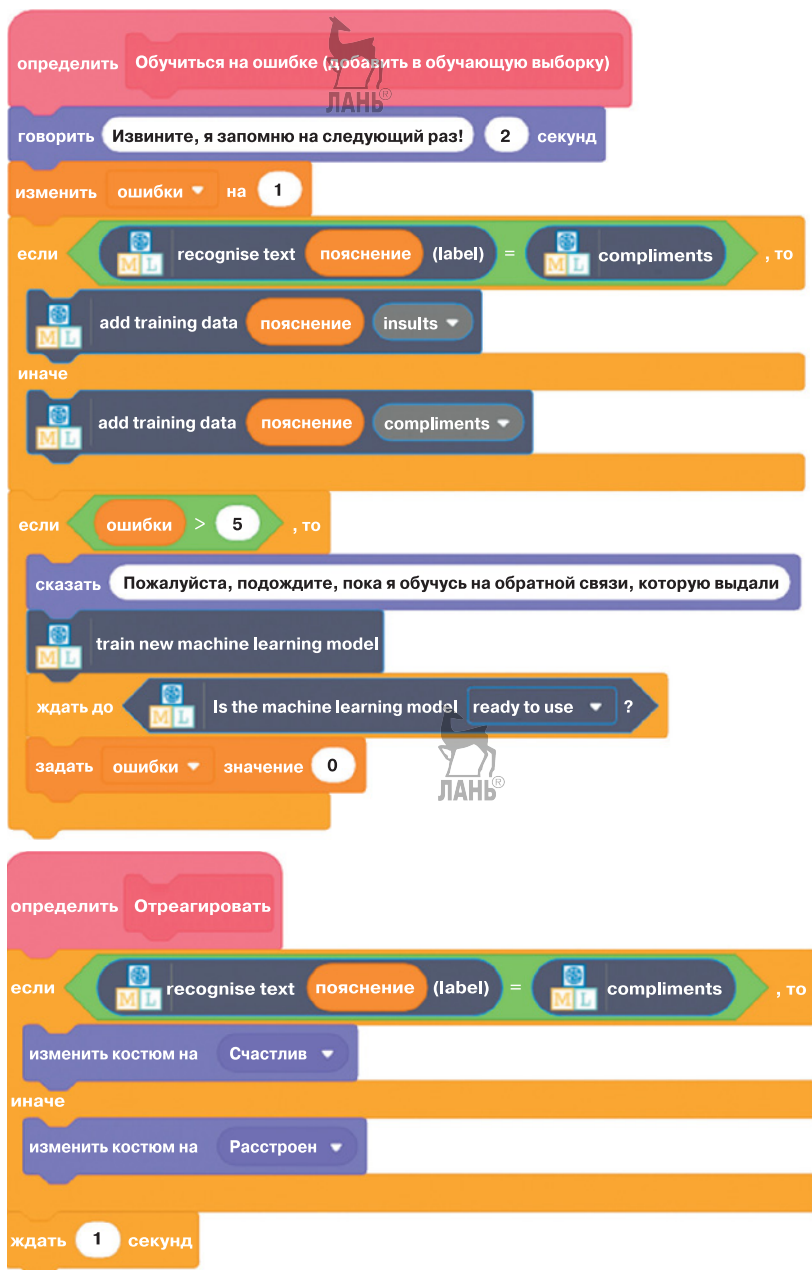


Рис. 7.23. Пример скрипта для обучения модели на ошибках (окончание)

Придумайте свой способ донесения до персонажа того, что он ошибся, и измените тогда скрипт, чтобы он мог учиться на ваших ответах.

Что вы узнали



В этой главе вы узнали о технологии анализа тональности текста — одном из способов использования машинного обучения для распознавания эмоций в тексте. Вы узнали, как компании могут использовать анализ тональности текста для получения и анализа обратной связи от клиентов в Интернете или для сортировки почты и корреспонденции по уровню удовлетворенности клиентов, от которых она получена.

Вы убедились, что машинное обучение — более легкий и эффективный способ создания систем искусственного интеллекта для больших проектов, чем написание пошаговых инструкций. Вы также узнали, как показатель степени уверенности может дать вам понять, насколько модель машинного обучения уверена в результате распознавания тестового примера. Еще вы увидели, как можно улучшить модель машинного обучения, если научить ее обучаться на своих ошибках. Это очень важно!

В следующей главе вы познакомитесь с новым подходом, который похож на анализ тональности текста. Вы научите модель машинного обучения распознавать разные стили письма.





ГЛАВА 8

Распознавание стиля письма в газетных статьях

В предыдущей главе вы узнали, что компьютер можно научить распознавать различные характеристики текста. Выполняя предыдущий проект, вы обучили свою модель распознавать смысл письма — комплименты и оскорбления. Но компьютеры могут распознавать и стиль письма.

Например, модель машинного обучения можно обучить распознавать различия между официальным письмом и неформальной перепиской. Делается это точно таким же образом — на большом количестве примеров модель нужно обучить распознавать закономерности в словах и фразах, которые люди используют в том или ином стиле письма.

Различные газеты и новостные сайты используют совершенно разные слова и фразы для описания одних и тех же событий. В этой главе вы обучите модель машинного обучения определять тип издания, из которого взят заголовок, как показано на рисунке 8.1.



Рис. 8.1. Распознавание заголовков газетных статей

Начнем!

Выполнение проекта



Цель этого проекта — выяснить, сможет ли компьютер научиться распознавать стили текста, которые используют разные издания. В качестве примера я обучил свою модель машинного обучения распознавать заголовки из *таблоидов* (небольших по формату газет со сплетнями и большим количеством фотографий) и *традиционных широкоформатных газет* с более официальным стилем.

Вы можете выполнить свой проект, ориентируясь на мой пример, или придумать другие критерии отбора. Если вы решите второе, обратите внимание, что вам следует выбрать только одну стилевую особенность, которую компьютер будет учиться распознавать. Например, вы можете провести распознавание по следующим характеристикам:

- **заголовки американских газет и заголовки британских газет.** Различается ли стиль текста и формулировки в американской и британской прессе?
- **заголовки газет, опубликованные летом, и заголовки газет, опубликованные зимой.** Зависит ли стиль заголовков от времени года?
- **общенациональные газеты и местные газеты.** Различается ли стиль заголовков у газет общенационального распространения и районных газет?
- **заголовки газет, опубликованные в выходные, и заголовки газет, опубликованные в будни.** Различаются ли стиль текста и формулировки в газетах, выпускаемых в рабочие (учебные) или нерабочие дни?
- **статьи, публикуемые двумя разными газетами.** Если вы выберете две конкретные газеты, можно ли обучить компьютер распознавать статьи и заголовки каждой из них?
- **заголовки газет, изданных в 1950-х годах, и заголовки современных газет.** Отличаются ли сегодняшние заголовки от заголовков в старых газетах?

Вам понадобятся примеры обоих типов газетных статей или заголовков, которые должна научиться распознавать ваша модель машинного обучения. Есть много веб-сайтов, которые могут помочь вам в этом*.

Подумайте, какая пара примеров для сравнения показалась вам интересной, а затем оцените, сможете ли вы найти достаточно примеров на выбранную тему.

* Например, вы можете использовать архив газеты «Пионерская правда»: <https://arch.rgdb.ru/xmlui/handle/123456789/39290> — Прим. ред.

Вам понадобятся две группы для обучения модели машинного обучения:

- примеры газетных статей или заголовков того типа, который должна научиться распознавать ваша модель машинного обучения;
- примеры газетных статей или заголовков, которые не попадают под заданные вами критерии.

Для второй группы постарайтесь изменить только одну характеристику. Например, если вы выбрали тему «Заголовки британских таблоидов 1970-х годов», то она содержит сразу три характеристики: страна (Великобритания), тип газеты (таблоид) и временной отрезок (1970-е годы). Тогда что можно выбрать для второй группы примеров?

Например, вы можете поместить в эту группу заголовки американских таблоидов 1970-х годов, тогда компьютер научится распознавать разницу между заголовками британских и американских таблоидов 1970-х годов. Или же вы можете заполнить ее примерами заголовков из современных британских таблоидов, чтобы компьютер научился распознавать разницу между заголовками британских таблоидов, опубликованных в 1970-х годах и сегодня.

Чего вам *не следует делать* в этом случае, так это использовать, например, заголовки из сегодняшних американских газет. Это изменит все три характеристики, из-за чего модели будет сложнее распознавать примеры, соответствующие одновременно нескольким из них.



Примечание



Выберите один критерий, который вы хотите, чтобы компьютер научился распознавать, и постарайтесь сохранить все остальные одинаковыми в обеих группах примеров.

Еще один пример. Если вы хотите научить компьютер распознавать, как написаны статьи в газете «Daily Times», во вторую группу примеров вы можете включить статьи из других газет, но за один и тот же день. Еще лучше, если вы попытаетесь подобрать статьи за тот же день на ту же тему. Тогда ваша модель машинного обучения будет учиться распознавать не тему статьи, а шаблоны стиля, присущего «Daily Times», для обсуждения одной и той же темы.

В моем примере для того, чтобы научить модель машинного обучения распознавать заголовки из таблоидов, я сравнил примеры заголовков таблоидов с заголовками из новостных ши-

рокоформатных газет. И вот какие характеристики я оставил одинаковыми:

- я использовал газеты, выпущенные в один и тот же период — с марта по апрель 2015 года;
- я использовал один и тот же раздел каждой газеты — заголовков самой большой статьи на первой странице газеты;
- я использовал однотипные газеты — общенациональные, выпущенные в будни;
- я использовал газеты одной и той же страны — Великобритании.

Обучение модели

1. Создайте новый проект машинного обучения и назовите его «Газеты». В качестве распознаваемого типа данных укажите «текст», после чего выберите язык, на котором будет написан распознаваемый текст, — «Another language» (англ.: другой язык).



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 8.2.

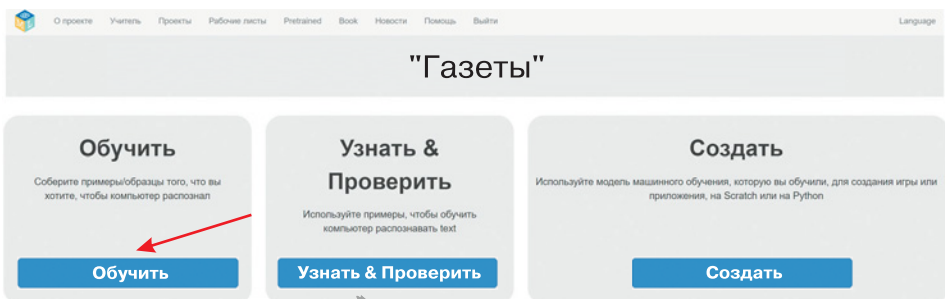


Рис. 8.2. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

3. Нажмите на кнопку «Добавить новую метку» (рис. 8.3) для создания коллекций двух групп, которые вы хотите сравнивать в вашем проекте. В моем проекте я назвал их «tabloid» (англ.: таблоид) и «broadsheet» (англ.: широкоформатная газета).

4. Найдите несколько примеров для первой группы и заполните ими первую коллекцию. Что это будут за примеры, зависит от выбранной вами темы проекта. Это могут быть газетные

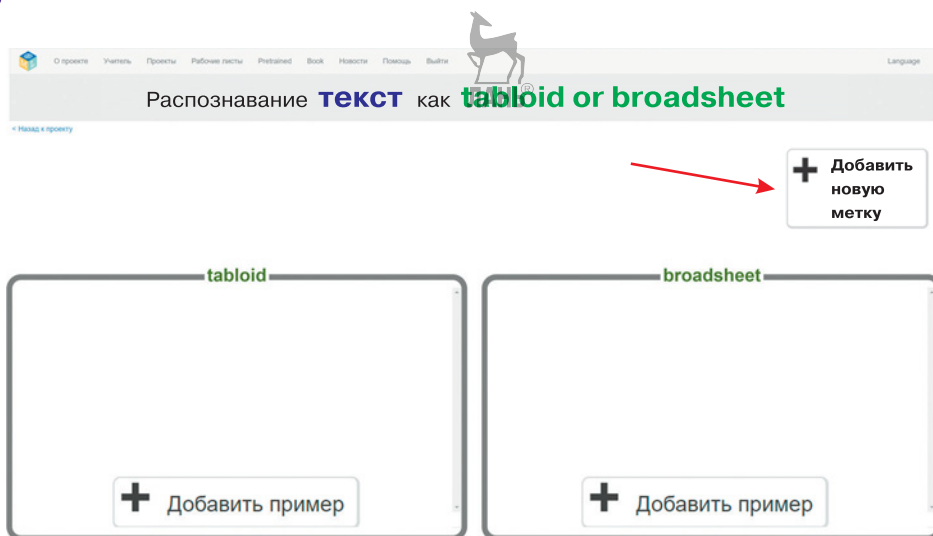


Рис. 8.3. Создание коллекций для двух групп примеров

заголовки, если вы делаете проект по заголовкам. А могут быть первые абзацы статьи, если вы сравниваете манеру изложения.

Для своего проекта я использовал заголовки газетных статей, взятые с веб-сайта, на котором размещены главные страницы национальных газет Великобритании. Вы тоже можете воспользоваться аналогичным ресурсом с архивом газет вашего региона.

5. Нажмите на кнопку «Добавить пример», чтобы добавить новый обучающий пример в коллекцию (рис. 8.4).

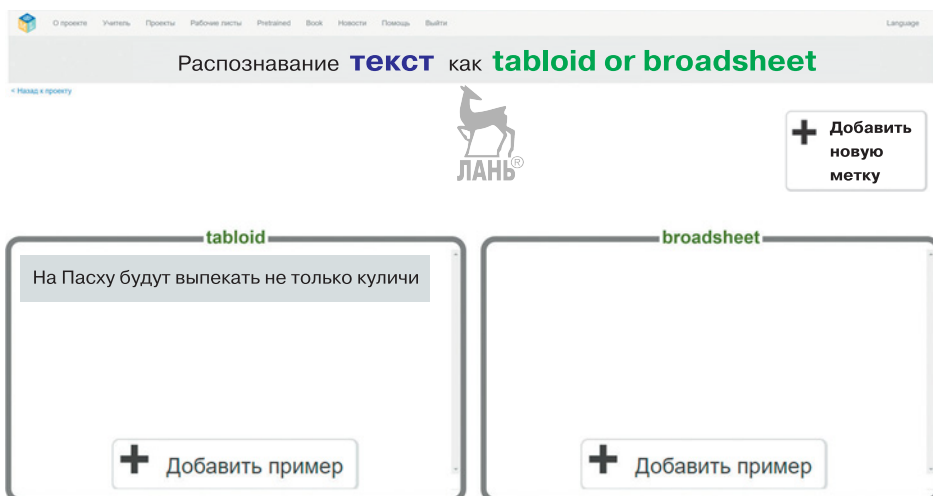


Рис. 8.4. Добавление первого обучающего примера в проект «Газеты»

6. Повторяйте шаги 4 и 5 до тех пор, пока не соберете как минимум 20 обучающих примеров в каждую из коллекций, как показано на рисунке 8.5.

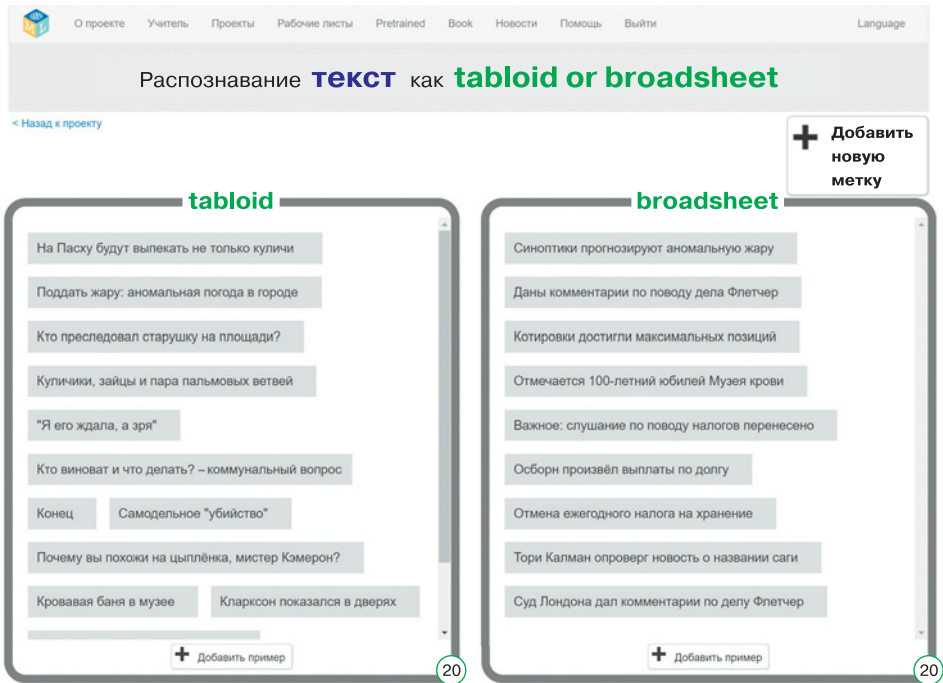


Рис. 8.5. Обучающие примеры для проекта «Газеты»

7. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

8. Нажмите на кнопку «Узнать и проверить», как показано на рисунке 8.6, чтобы перейти к следующему этапу проекта.

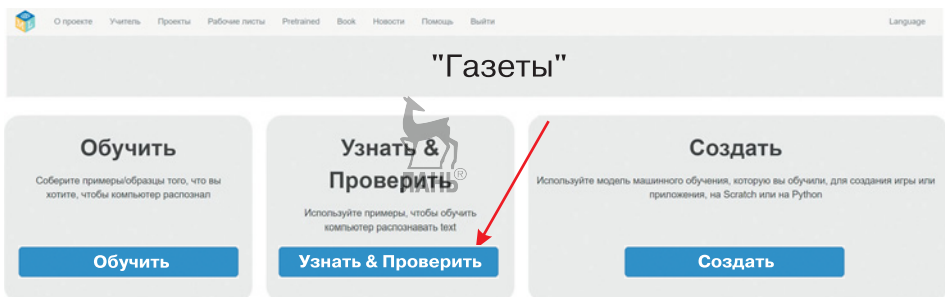


Рис. 8.6. Переход ко второму этапу работы над проектом — этапу «Узнать и проверить»

9. Нажмите на кнопку «Обучить новую модель машинного обучения», как показано на рисунке 8.7.

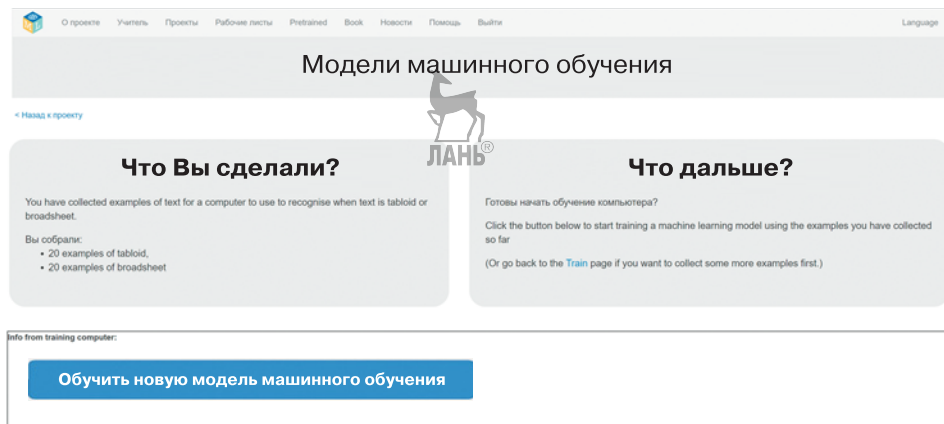


Рис. 8.7. Нажмите на кнопку «Обучить новую модель машинного обучения», чтобы начать обучение

Компьютеру может потребоваться несколько минут, чтобы изучить примеры, которые вы для него собрали. Пока вы ждете, задайтесь следующими вопросами:

- Заметили ли вы какие-либо общие закономерности, присущие каждой группе, когда добавляли их в коллекции?
- Есть ли такие темы, которые чаще освещаются в одной группе, чем в другой?
- Существуют ли определенные слова, термины или фразы, которые в одной группе используются чаще, чем в другой?
- Есть ли отличия в способах формулирования предложений? Например, в одной группе более длинные предложения, чем в другой. Или в какой-то из групп гораздо чаще мелькают заглавные буквы.
- Прослеживаются ли закономерности в типах используемых слов? Например, использует ли одна группа более простой язык или более короткие слова? Встречаются ли в одной группе более эмоционально окрашенные слова и выражения?

Попробуйте угадать, какие закономерности может определить и усвоить ваша модель машинного обучения из обучающих примеров, которые вы для нее подготовили.

Подготовка проекта

В проектах, которые вы делали до сих пор, вы тестировали свои модели машинного обучения, опробовав их на практике. В этой главе вы познакомитесь с несколькими способами *формального*

тестирования проектов машинного обучения, которые используются в реальном мире. Это несложно!

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

2. Нажмите на кнопку «Создать».

3. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch.

4. Удалите спрайт кота (Спрайт 1), поскольку в этом проекте он вам не понадобится. Наведите на него указатель мыши и нажмите на иконку корзины, которая появится в правом верхнем углу спрайта. Вкладка «Костюмы» теперь должна быть заменена вкладкой «Фоны».

5. Перейдите на вкладку «Фоны» и нарисуйте (или загрузите) фон, который поделит ваш проект Scratch на две части, как показано на рисунке 8.8.

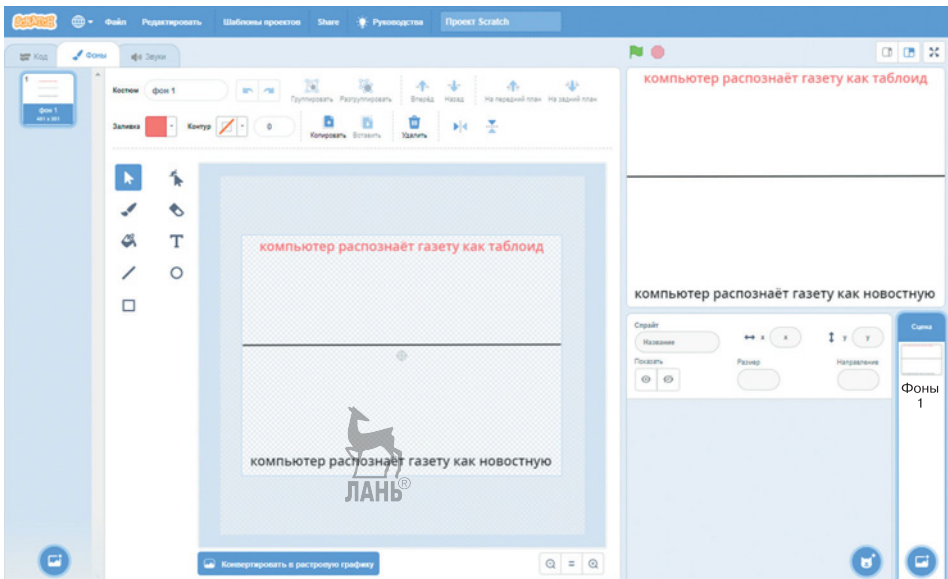


Рис. 8.8. Подготовьте фон для проекта «Газеты»

Верхняя часть будет предназначена для газет, соответствующих характеристикам, которые научилась распознавать ваша модель машинного обучения, а нижняя часть — для газет, которые не соответствуют этим характеристикам.

Выберите свой цвет шрифта для каждой половины. Я использовал красный для верхней половины и черный для нижней половины.

Нарисуйте линию, чтобы визуальнo разделить пространство на две половины.

У вас должно получиться что-то похожее на рисунок 8.8.

6. Добавьте в проект новый спрайт, наведя указатель мыши на значок «Выбрать спрайт» (кошачья мордочка) в правом нижнем углу экрана. Этот спрайт будет представлять собой заголовки газет, которые должна распознавать ваша модель машинного обучения. Поэтому дайте этому спрайту подходящее имя как в текстовом поле «Костюм» над холстом, так и в текстовом поле «Спрайт» в правом нижнем углу.

Чтобы нарисовать ваш собственный значок для газеты, нажмите на кнопку «Нарисовать». Или же, чтобы загрузить уже готовое изображение с вашего компьютера, нажмите на кнопку «Загрузить спрайт».

7. Перейдите на вкладку «Костюмы» и используйте различные инструменты для рисования, чтобы раскрасить газету так, чтобы она соответствовала верхней части фона.

В своем примере я нарисовал красную газету и дал этому спрайту имя «tabloid», как показано на рисунке 8.9.

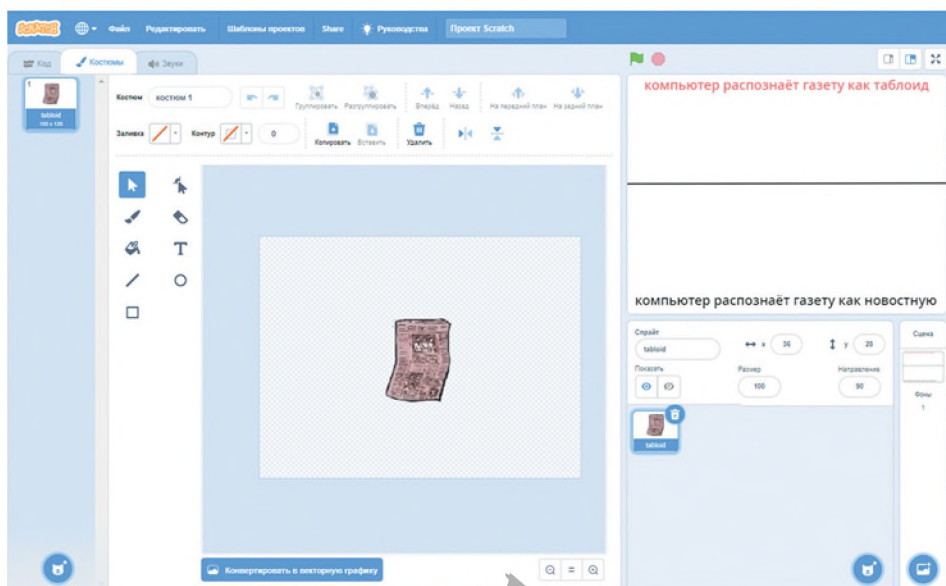


Рис. 8.9. Нарисуйте спрайт для газет первого типа

8. Перейдите на вкладку «Код», выберите категорию «Переменные» в Панели инструментов и нажмите на кнопку «Создать список» (рис. 8.10).

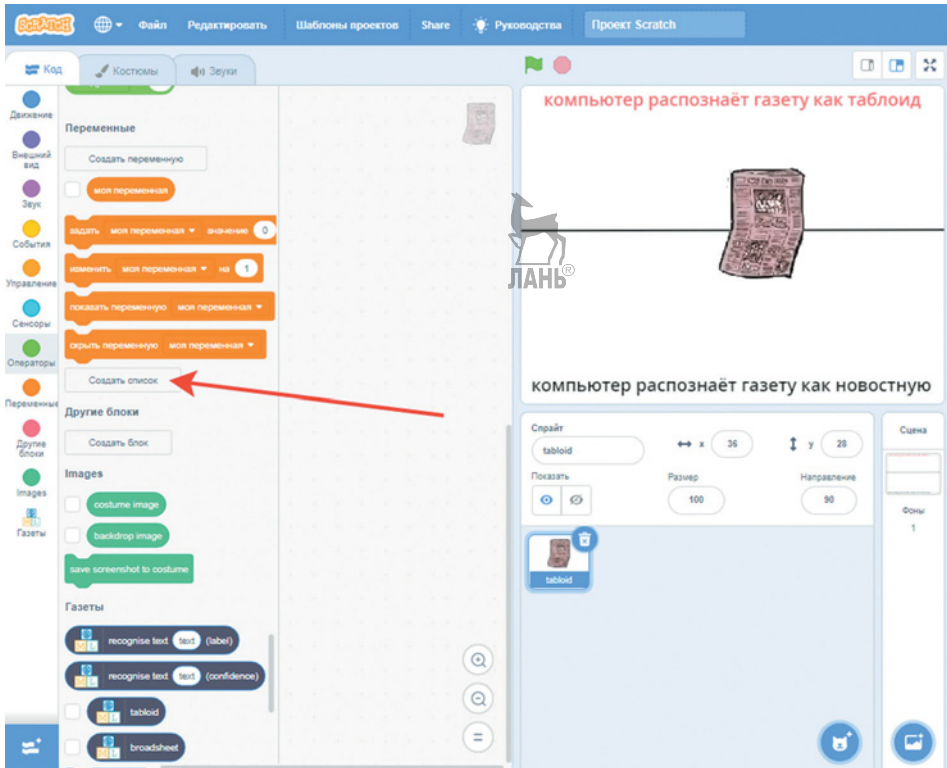


Рис. 8.10. Создайте список для газетных заголовков первого типа

Поставьте галочку в чекбокс «для всех спрайтов» и дайте списку имя, которое будет соответствовать характеристикам, которые вы научили распознавать свою модель машинного обучения. В своем примере я назвал список «Заголовки таблоидов».

9. Наберите на клавиатуре текст и добавьте в список не менее 10 примеров, как показано на рисунке 8.11.

Нажмите кнопку «+» в левом нижнем углу списка, чтобы добавить новую строку. Вы можете перетащить знак «=» из правого нижнего угла, чтобы увеличить ширину списка и полностью видеть то, что вы печатаете.

Текст, который вы вводите в этот список, представляет собой примеры газетных заголовков или статей, соответствующих тем характеристикам, которые ваша модель машинного обучения научилась распознавать.

Важно, чтобы это были новые заголовки или статьи, которые вы еще не использовали в качестве обучающих примеров. Это должны быть заголовки или статьи, которые компьютер раньше не видел, чтобы вы могли по-настоящему проверить, научил-

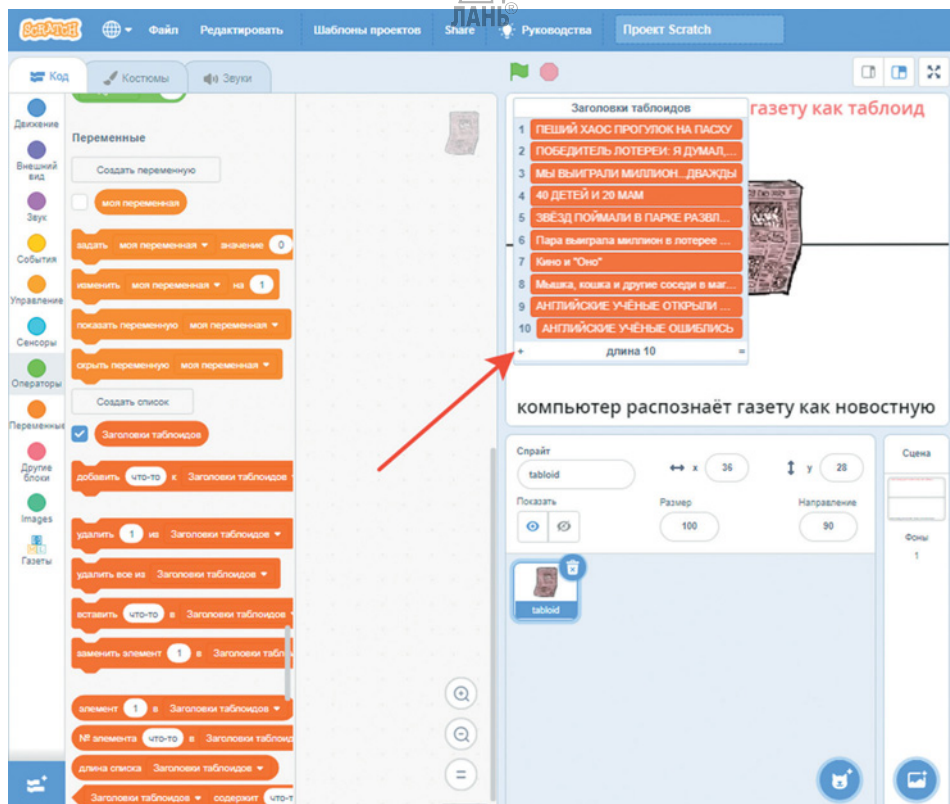


Рис. 8.11. Тестовые примеры для первой группы

ся ли он действительно распознавать, а не просто запоминать примеры.

Когда вы закончите вводить новые примеры в список, снимите галочку из чекбокса со списка на Панели инструментов, чтобы скрыть его из рабочей области.

10. Перейдите на вкладку «Код» и создайте такой же скрипт, как показан на рисунке 8.12.

Измените название блока «Заголовки таблоидов» на название характеристики, которую вы научили распознавать компьютер.

Этот скрипт будет проходить через каждый из моих заголовков-примеров, и если модель машинного обучения распознает его как заголовок таблоида, то переместит этот заголовок в верхнюю половину экрана. Иначе заголовок переместится в нижнюю половину экрана.

Вы уже встречались с чем-то подобным в главе 3, когда сортировали спрайты животных с помощью модели машинного обучения, распознающей спрайт по костюму. На этот раз вы со-

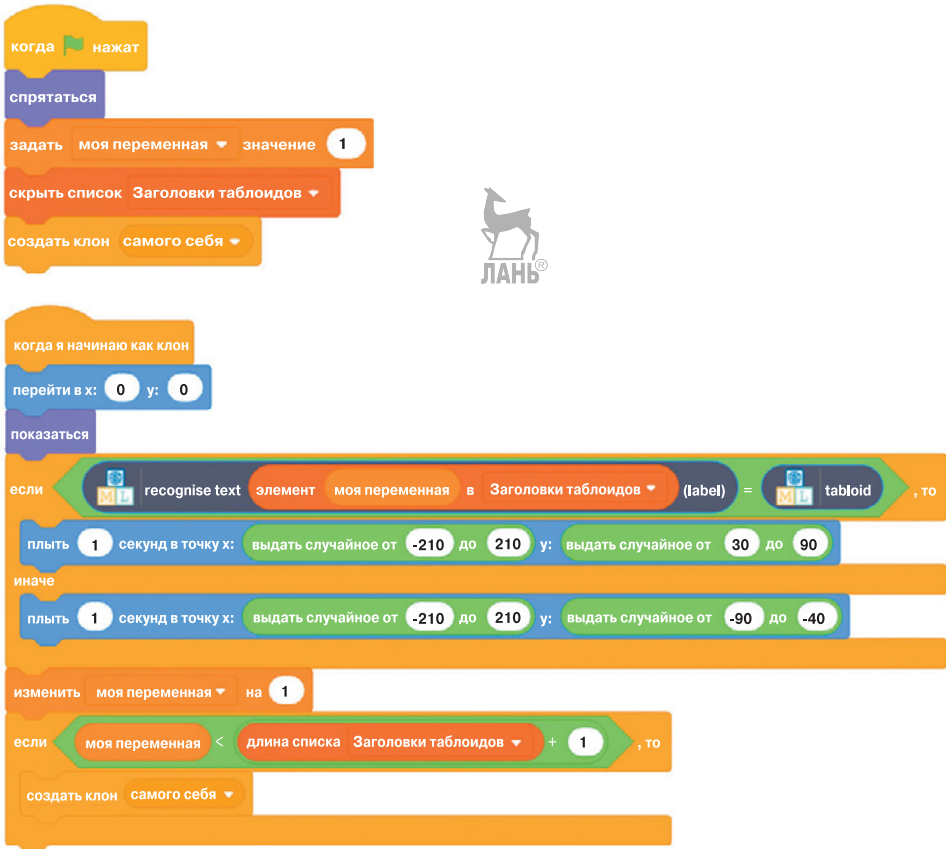


Рис. 8.12. Первый фрагмент скрипта для проекта «Газеты»

ртируете спрайты с помощью модели машинного обучения, которая распознает некоторый текст в вашем списке.

11. Нажмите на зеленый флажок, чтобы запустить и протестировать скрипт.

Насколько успешно справилась ваша модель машинного обучения? Возможно, она допустила несколько ошибок, но в целом должна справляться хорошо. На рисунке 8.13 вы можете увидеть, каких успехов достигла моя модель.

Однако по одной только этой группе газетных заголовков вы не можете с уверенностью заявить, способна ли моя модель машинного обучения отличать заголовки таблоидов от заголовков широкоформатных газет. Может быть, она распознает все что угодно как заголовки таблоидов. Мне нужно добавить несколько примеров заголовков широкоформатных газет в свой список, чтобы узнать, распознает ли их моя модель машинного обучения.

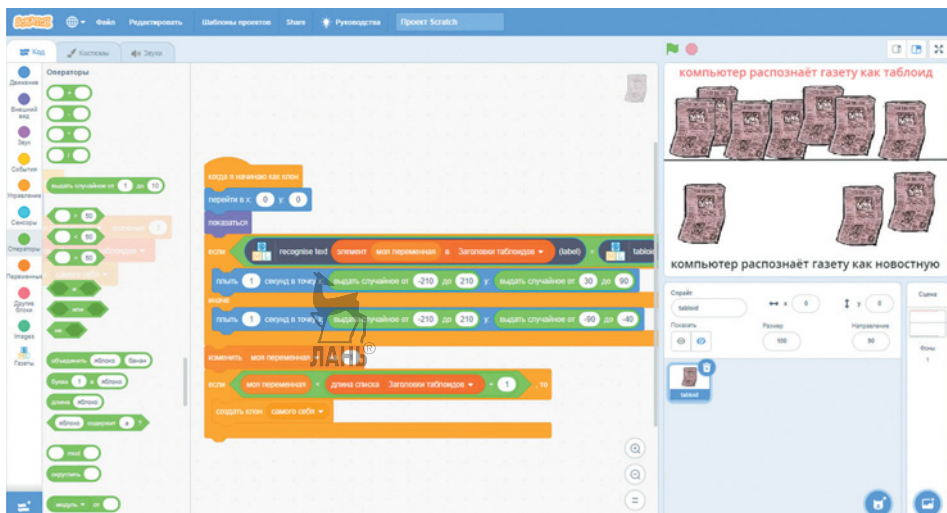


Рис. 8.13. Первое тестирование проекта «Газеты»

12. Следуйте инструкциям из шага 6, чтобы создать еще один спрайт, представляющий собой вторую группу газет, которые должны после распознавания отправляться в нижнюю часть экрана. Раскрасьте этот спрайт в тот же цвет, что и у изображения нижней половины фона. Задайте этому спрайту соответствующее имя.

Для своего примера я нарисовал синюю газету, обозначающую заголовки широкоформатных газет (рис. 8.14).

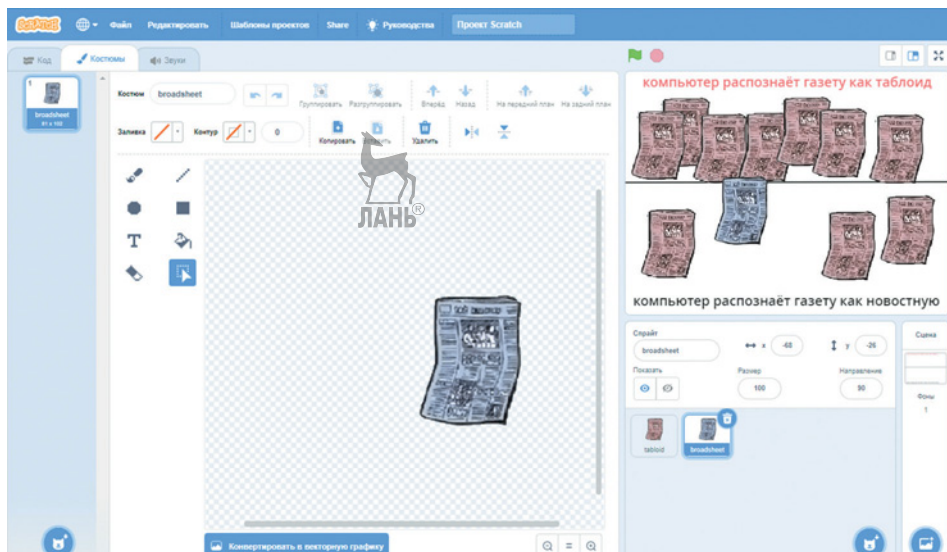


Рис. 8.14. Второй спрайт для проекта «Газеты»

13. Перейдите на вкладку «Код», выберите категорию «Переменные» в Панели инструментов, а затем нажмите на кнопку «Создать список». Заполните этот список как минимум 10 примерами заголовков или статей, которые после распознавания должны отправляться в нижнюю часть экрана.

Как и в прошлый раз, убедитесь, что в чекбоксе «для всех спрайтов» установлена галочка, и назовите список в соответствии с распознаваемыми характеристиками.

Для моего проекта я создал список «Заголовки широкоформатных газет» и добавил в него 10 примеров заголовков широкоформатных газет (рис. 8.15).

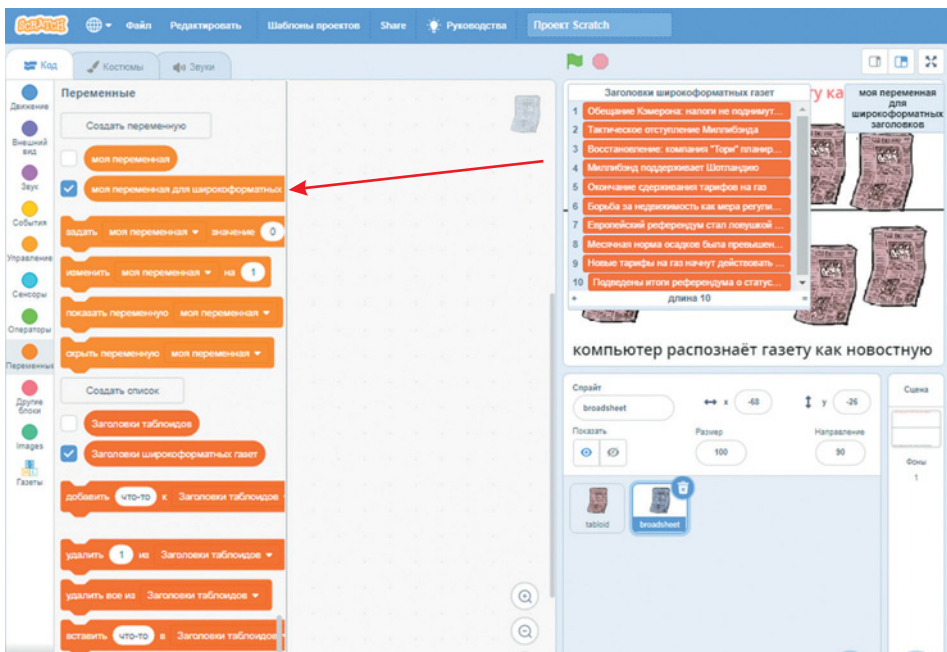


Рис. 8.15. Подготовка тестовых примеров для второй группы газет

После того как вы добавите нужное количество примеров, снимите галочку из чекбокса рядом со списком на Панели инструментов, чтобы он не отображался на Сцене.

14. Создайте новую переменную, которая будет считать распознанные заголовки. Для этого выберите категорию «Переменные» в Панели инструментов и нажмите на кнопку «Создать переменную». Назовите переменную так же, как и вторую группу распознаваемых газет.

В моем примере я назвал переменную «Моя переменная для широкоформатных заголовков» (см. рис. 8.15).

15. Для второго спрайта создайте скрипт как на рисунке 8.16.

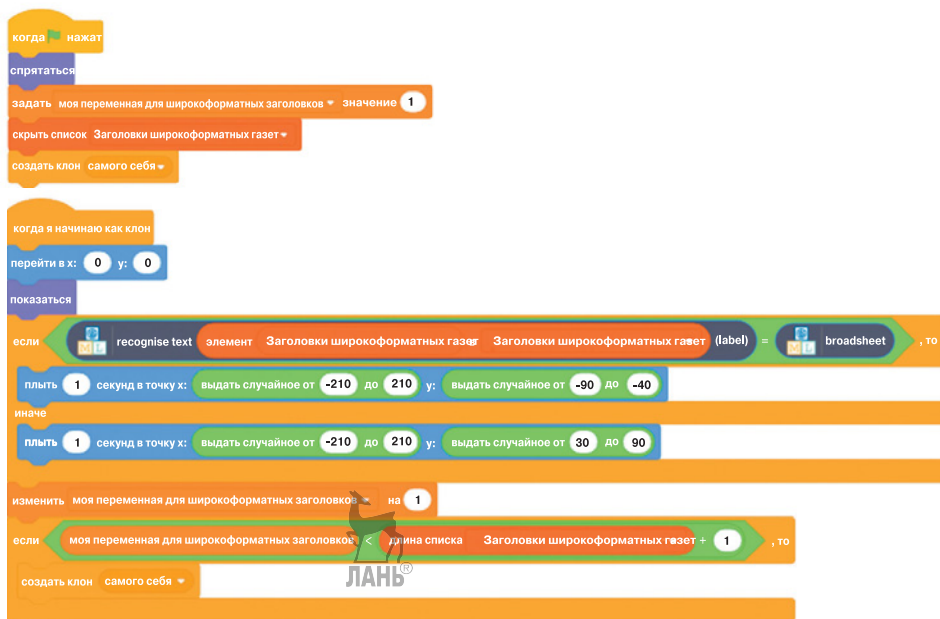


Рис. 8.16. Скрипт для повторного тестирования проекта «Газеты»

Эти фрагменты скрипта очень похожи на те, что вы уже создали для первого спрайта на шаге 9. Скрипт будет проходить через тестовые примеры во втором списке, который вы создали. У каждого из них будет свой спрайт, и, если ваша модель машинного обучения распознает пример как подходящий для второй группы, он будет перемещен в нижнюю половину экрана. Иначе скрипт переместит его в верхнюю половину экрана.

Будьте осторожны и не перепутайте имена списков и переменных, чтобы не использовать здесь список или переменную, которые вы создали для вашего первого спрайта.

Мой первый скрипт был о заголовках таблоидов, а этот — о заголовках широкоформатных газет.



Примечание

Убедитесь, что ваши скрипты отличаются и соответствуют спрайтам, для которых они созданы.

Обратите внимание, что координаты в этом скрипте также должны быть другими (не такими, как в скрипте для первого спрайта), чтобы на этот раз в случае совпадения примеры попадали в нижнюю половину экрана (а не в верхнюю, как в первом скрипте).

16. Снова нажмите на зеленый флажок, чтобы запустить скрипт.

Как и в прошлый раз, ваша модель машинного обучения иногда ошибается, это нормально. На рисунке 8.17 вы можете увидеть, как справилась моя модель машинного обучения во второй раз.

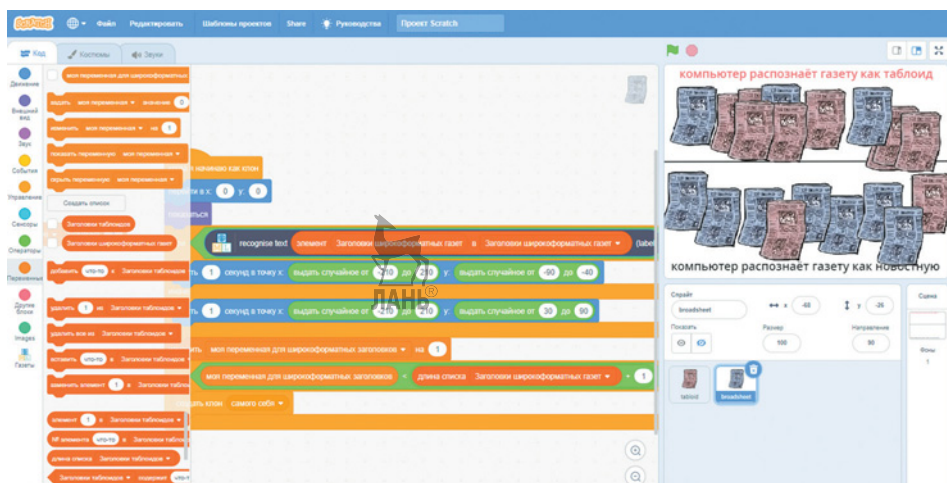


Рис. 8.17. Результаты повторного тестирования проекта «Газеты»

А насколько успешно справилась ваша модель?

Обзор и улучшение проекта

Вы обучили модель машинного обучения распознаванию стиля письма в газетных статьях! Вы также использовали Scratch, чтобы протестировать проект и визуализировать эффективность вашей модели машинного обучения.

Если бы моя модель машинного обучения работала идеально, она переместила бы все красные газеты в верхнюю половину экрана, а все синие газеты в нижнюю половину экрана. Но это не так.

Абсолютно правильное распознавание заголовков в данном случае маловероятно, так как я обучил свою модель на небольшом количестве примеров. Если бы я использовал больше обучающих примеров, результаты могли бы быть лучше, но даже в этом случае системы машинного обучения редко работают идеально.

Но как же понять, насколько хорошо модель машинного обучения справляется со своей задачей?

Есть много способов, которыми мы можем описать производительность системы машинного обучения. Рассмотрим несколько примеров.

Оценка качества модели машинного обучения: показатель достоверности

Одной из характеристик, которые используются для описания того, насколько хорошо работает модель машинного обучения, является **доля правильных ответов**, или **показатель достоверности** (англ.: ассигасу), или подсчет количества тестовых примеров, на которые модель машинного обучения дает правильные ответы.

Создайте новую переменную с именем «верные ответы», установив галочку в чекбоксе «для всех выбранных спрайтов».

Измените фрагменты скрипта для первого спрайта (тот, что показан на рисунке 8.12 для газет, которые после распознавания

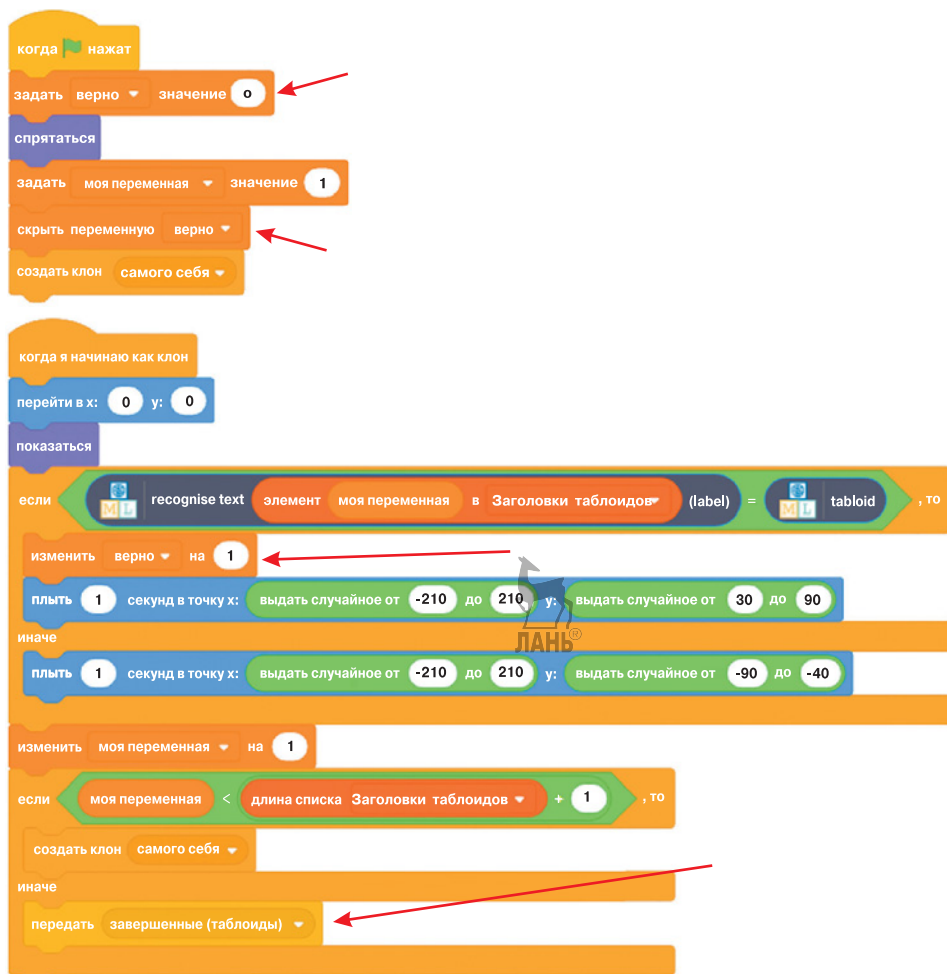


Рис. 8.18. Измените скрипт для первого спрайта (см. рис. 8.12), чтобы он научился подсчитывать количество верных ответов



должны помещаться в верхнюю половину экрана), чтобы они соответствовали фрагментам скрипта на рисунке 8.18.

Обратите внимание, что вам нужно будет заменить последний блок «Если ...» в скрипте на блок «Если ... то...».



Примечание

Вам не нужно воспроизводить комментарии в желтых полях. Я добавил их как подсказки, чтобы вам было проще увидеть, какие части скрипта изменились по сравнению с предыдущим вариантом.

Теперь измените фрагменты скрипта для второго спрайта (того, что представлен на рисунке 8.16 и предназначен для перемещения тестовых примеров в нижнюю часть экрана). Они должны соответствовать рисунку 8.19.

Обратите внимание, что вам нужно будет заменить последний блок «Если ...» в скрипте на блок «Если ... то ...».



Примечание

Как и в прошлый раз, вам не нужно воспроизводить комментарии, они нужны только для того, чтобы помочь вам понять, что именно нужно изменить.

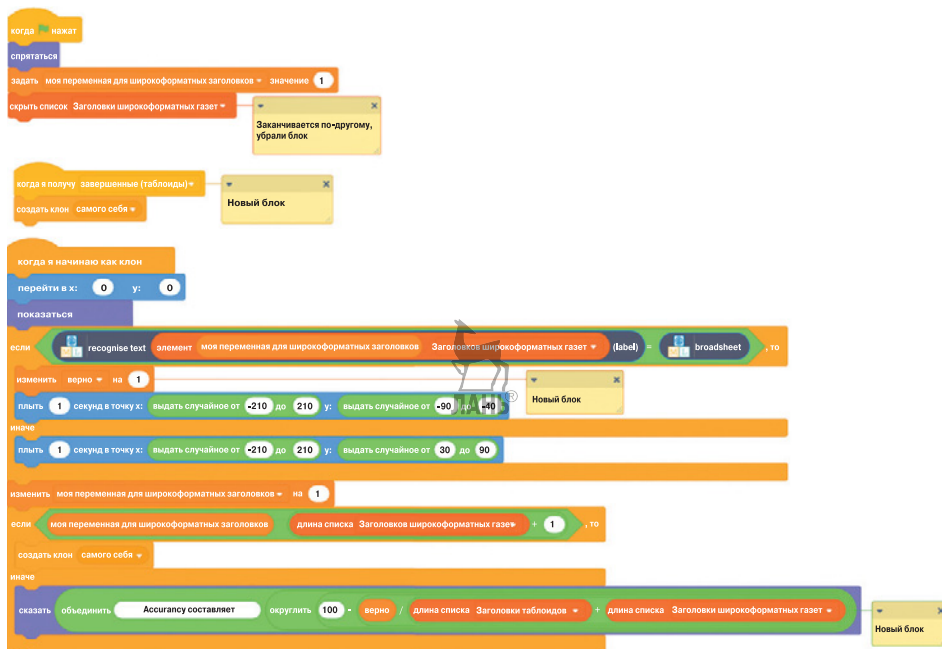


Рис. 8.19. Измените скрипт для второго спрайта (см. рис. 8.16), чтобы он научился считать количество верных ответов

После того как вы измените скрипты, ваш проект Scratch сможет определить **показатель достоверности (accuracy)** используемой модели машинного обучения и выдать результаты в конце тестирования. Формула для расчета:

Количество верных ответов / [(количество заголовков таблоидов) + (количество заголовков широкоформатных газет)]

Скрипт для этой формулы показан на рисунке 8.20.

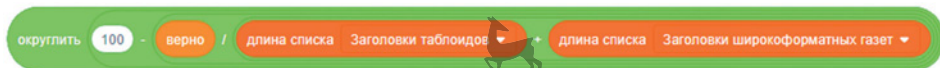


Рис. 8.20. Вычисление показателя достоверности модели машинного обучения (Scratch определяет количество элементов в списке как длину списка)

Другими словами, показатель достоверности отражает процент того, сколько заголовков (от общего количества) правильно распознала модель машинного обучения.

Достоверность моей модели машинного обучения иллюстрирует рисунок 8.21. А какой процент достоверности дает ваша модель машинного обучения?



Рис. 8.21. Отображение показателя достоверности модели машинного обучения в конце тестирования проекта

Оценка качества модели машинного обучения: матрица ошибок

Достоверность (ассигансу) — полезный показатель для оценки качества моделей машинного обучения и, пожалуй, самый известный.

Но часто только его одного недостаточно, чтобы охарактеризовать работоспособность реальных систем машинного обучения.

Как вы думаете, могут ли возникнуть какие-либо ошибки и проблемы с показателем достоверности?

Что, если бы моя модель машинного обучения думала, что все тестовые примеры — это заголовки из новостных широкоформатных газет, как показано на рисунке 8.22?



Рис. 8.22. Модель машинного обучения распознала все заголовки как заголовки широкоформатных газет

В этом случае показатель достоверности будет составлять 50%, так как модель машинного обучения поместит все 10 новостных газет в нужное место (т. е. она распределит все варианты в одну из половин экрана), но все 10 таблоидов — в неправильное.

Но «показатель достоверности 50%» — плохое описание системы, которая дает один и тот же ответ на любой вопрос. Нам нужен лучший способ описать качество нашей модели машинного обучения, который позволит избежать подобной двусмысленности.

Матрица ошибок — это инструмент, в котором вы подсчитываете количество тестовых примеров, распознанных моделью

машинного обучения правильно, и количество тестовых примеров, распознанных неправильно, а затем упорядочиваете эти результаты в таблице. Давайте рассмотрим это на примере.

Создайте четыре новые переменные (не забудьте установить галочку в чекбоксе «для всех спрайтов») и назовите их «истинно положительные», «истинно отрицательные», «ложноположительные» и «ложноотрицательные», как показано на рисунке 8.23. Также убедитесь, что для всех переменных установлены галочки в чекбоксах на Панели инструментов.



Рис. 8.23. Подготовка переменных для создания матрицы ошибок

Расположите эти переменные на Сцене так, как показано на рисунке 8.24.

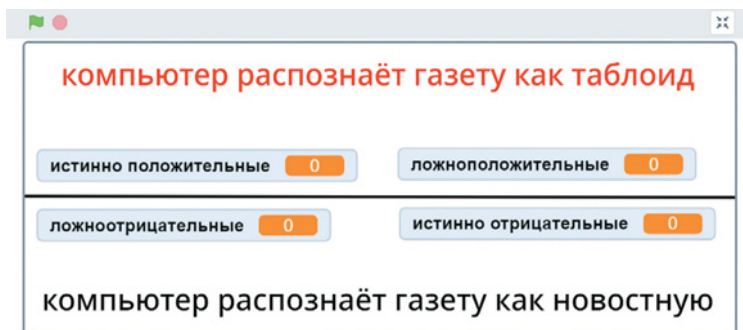


Рис. 8.24. Создание матрицы ошибок в Scratch

Измените свой скрипт для первого спрайта (тот, что показан на рисунке 8.18) так, чтобы он соответствовал рисунку 8.25.

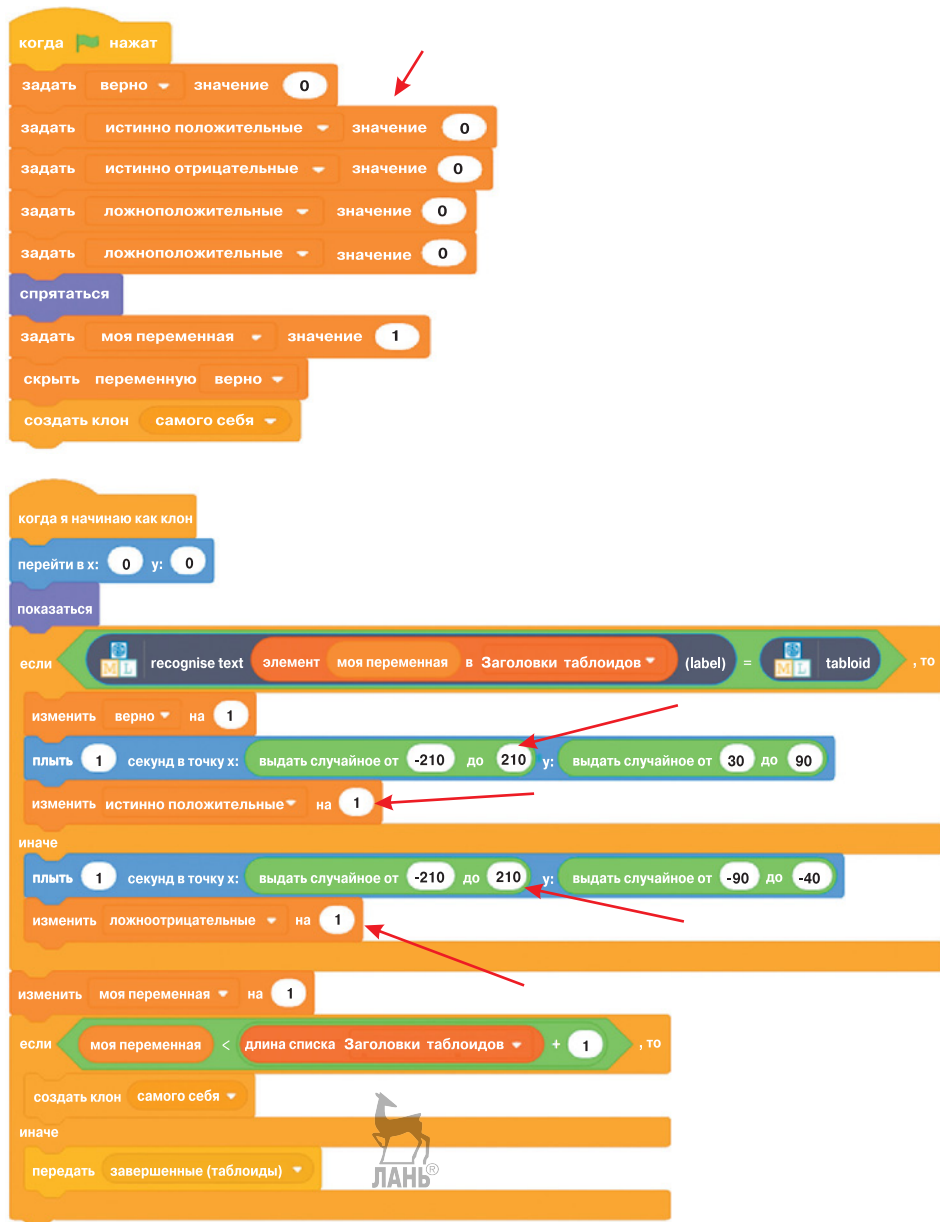


Рис. 8.25. Измените скрипт для первого спрайта (см. рис. 8.18), чтобы добавить вычисление значений для матрицы ошибок

Этот скрипт гораздо длиннее, поэтому не спешите и внимательно сверяйте свой скрипт с рисунком в книге. Как и в прошлый раз, я добавил комментарий, чтобы вам было легче отследить изменения.

В этом скрипте мы увеличиваем значение счетчиков истинно положительных и ложноотрицательных результатов и меняем значения x для перемещения блоков, чтобы таблоиды в любом случае оставались в левой части экрана.

Теперь измените скрипт для второго спрайта (того, что показан на рисунке 8.19) так, чтобы он соответствовал рисунку 8.26.

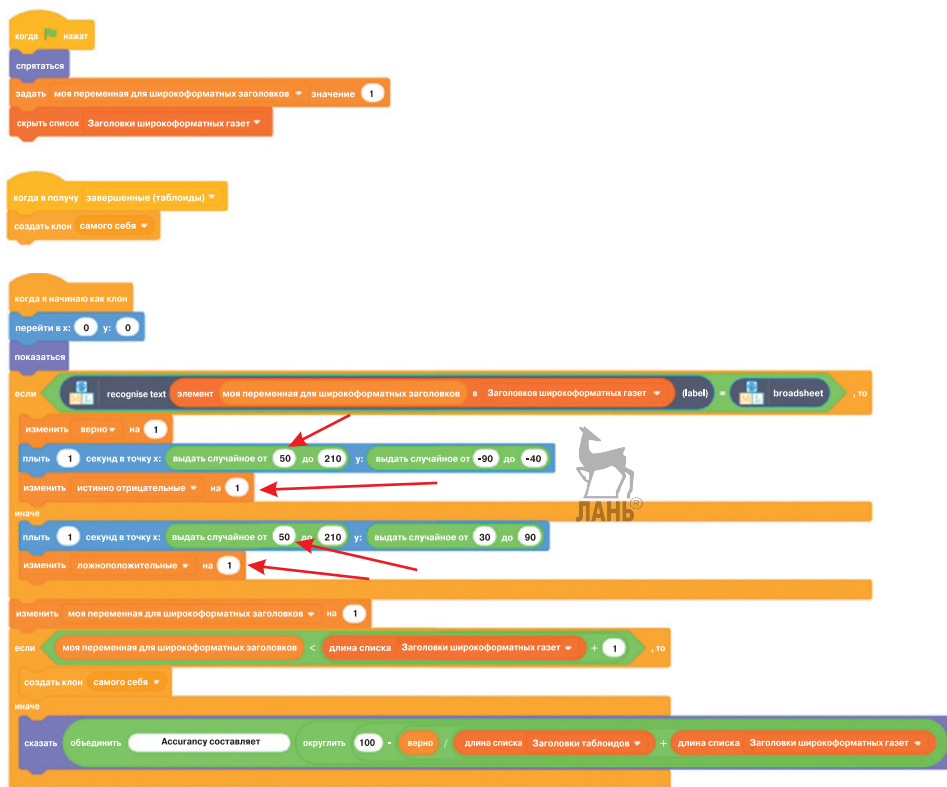


Рис. 8.26. Измените скрипт для второго спрайта (см. рис. 8.19), чтобы добавить вычисление значений для матрицы ошибок

В этом скрипте мы увеличиваем значение счетчиков ложноположительных и истинно отрицательных результатов и меняем значения x для перемещения блоков, чтобы широкоформатные газеты в любом случае оставались в правой части экрана.

Когда закончите, снова нажмите на зеленый флажок, чтобы протестировать проект после внесения изменений. Мои результаты показаны на рисунке 8.27.



Рис. 8.27. Отображение значений матрицы ошибок в Scratch

В таблице 8.1 показаны результаты заполнения матрицы ошибок для моей модели машинного обучения.

Таблица 8.1. Результаты заполнения матрицы машинного обучения

Истинно положительные В тестовом примере заголовок был из таблоида. Модель машинного обучения определила, что заголовок из таблоида. (Это правильный ответ)	Ложноположительные В тестовом примере заголовок был из широкоформатной новостной газеты. Модель машинного обучения определила, что заголовок из таблоида. (Это неправильный ответ)
Истинно отрицательные В тестовом примере заголовок был из таблоида. Модель машинного обучения определила, что заголовок из широкоформатной газеты. (Это неправильный ответ)	Ложноотрицательные В тестовом примере заголовок был из широкоформатной новостной газеты. Модель машинного обучения определила, что заголовок из широкоформатной газеты. (Это правильный ответ)

На самом деле построение матрицы ошибок — это полезный способ описать качество модели машинного обучения, и этот показатель говорит нам больше, чем может дать показатель достоверности (если он используется сам по себе).

Если бы моя модель машинного обучения думала, что все тестовые примеры — из широкоформатных новостных газет, тогда матрица ошибок выглядела бы так, как показано на рисунке 8.28.



Рис. 8.28. Пример матрицы ошибок

Оценка качества модели машинного обучения: точность и полнота

Еще один способ, которым мы можем описать качество моделей машинного обучения, — использование показателей *точности* (*precision*) и *полноты* (*recall*).

Точность рассчитывается как **число истинно положительных результатов / (число истинно положительных + число ложноположительных)**.

Полнота рассчитывается как **число истинно положительных результатов / (число истинно положительных + число ложноотрицательных)**.

Обновите свой скрипт, включив в него расчеты, показанные на рисунке 8.29. Добавьте их в конец блока «когда я запускаю клонирование» для второго спрайта после блока «говорить».



Рис. 8.29. Вычисление точности и полноты в Scratch

Я добавил эти вычисления в свой пример. На рисунке 8.30 вы можете увидеть, что у меня получилось.



Рис. 8.30. Отображение значений показателей точности и полноты в Scratch

Оценка точности означает, что, когда моя модель машинного обучения думает, что что-то является заголовком таблоида, это верно в 78% случаев.

Оценка полноты означает, что в тестовых примерах найдено 70% заголовков таблоидов.

Оценка достоверности означает, что 75% ответов, которые дала модель машинного обучения, были правильными.

Точность, полнота и процент достоверности — все эти показатели помогают оценить качество модели машинного обучения.

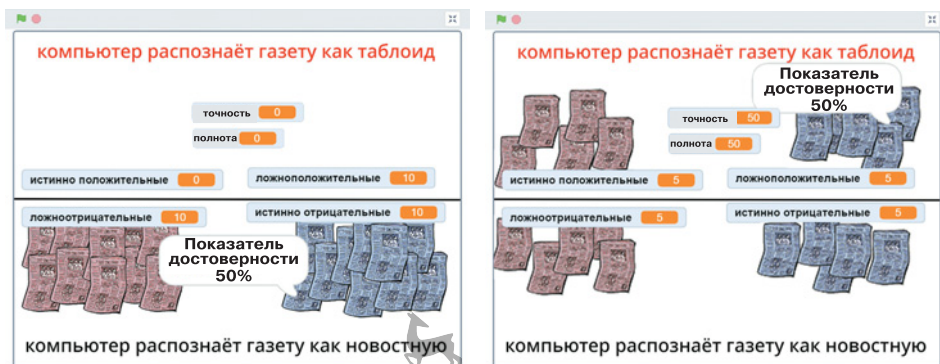


Рис. 8.31. Две разные модели машинного обучения с одинаковым показателем достоверности — 50%

Например, на рисунке 8.31 показаны результаты работы двух разных моделей машинного обучения, у обеих из которых показатель достоверности составляет 50%. Остальные показатели помогают описать, насколько различны эти модели в своей работе.

По сравнению с тем, как вы тестировали ваш проект в предыдущей главе (когда вы вручную вводили тестовые значения и получали представление о том, сколько раз модель отработала верно), использование численных показателей является более точным и эффективным способом описания качества модели машинного обучения. В следующей главе мы рассмотрим еще один способ оценки качества для проектов распознавания изображений.

Улучшение вашей модели машинного обучения

Попробуйте добавить еще по 10 примеров в каждую из коллекций с обучающими примерами на этапе обучения и используйте их для обучения новой модели машинного обучения.

После завершения обучения новой модели повторно запустите скрипт Scratch.

Как увеличение количества обучающих примеров повлияет на ваши показатели точности, полноты и достоверности?

Что вы узнали

Выполняя свои собственные проекты, вы могли заметить, что машинное обучение работает не идеально и может допускать ошибки. Однако для того чтобы быть полезным, машинному обучению вовсе не обязательно быть идеальным.

Например, системы машинного обучения компенсируют свои ошибки скоростью работы. Они могут за считанные минуты анализировать огромные объемы текста, на чтение которых у человека ушла бы целая жизнь. Даже если система машинного обучения делает ошибки в 10% случаев, она все равно может быть полезна при поиске вещей, требующих особого внимания и концентрации.

Но чтобы понимать, насколько хорошо работает система машинного обучения, важно знать, сколько ошибок она допускает.

Из этой главы вы узнали, что существует множество способов тестирования систем машинного обучения и измерения их качества. Эти знания должны помочь вам решить, как использовать полученные результаты.

В следующей главе вы узнаете, как решать более сложные проблемы, разбивая их на отдельные части, и вы снова будете использовать матрицу ошибок, чтобы понять, насколько хорошо работает ваша модель машинного обучения.





ГЛАВА 9

Поиск объекта на картинке



В предыдущих главах вы узнали, что модель машинного обучения можно научить распознавать объекты на картинке. Например, как это было, когда вы создавали игру «Камень, ножницы, бумага» в главе 4. В этой игре ваша рука, показывающая жесты, обозначающие камень, ножницы или бумагу, являлась единственным предметом на фотографии и больше на ней ничего не было. Но иногда мы хотим, чтобы компьютер научился находить что-то, что является лишь частью изображения. В этой главе вы увидите, как разбить сложный объект на отдельные части, а затем использовать машинное обучение для каждой из частей.

Например, представьте себе, что вам нужно с помощью машинного обучения определить, есть ли дерево на рисунке 9.1.

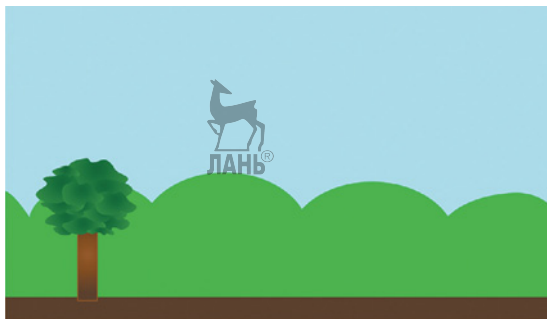


Рис. 9.1. Есть ли дерево на этом рисунке?

Основная идея этого проекта заключается в том, что вы обучаете свою модель машинного обучения распознавать изображения деревьев так же, как вы обучали ее распознавать изображения определенных животных в главе 3. Затем вы нарезаете фотографии на более мелкие части и используете свою модель машинного обучения, чтобы проверить, похожа ли какая-то из частей фотографии на изображение дерева.

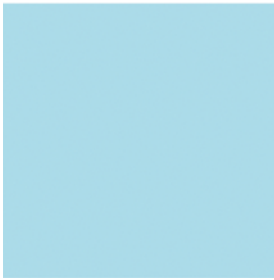


Рис. 9.2. Фрагмент рисунка 9.1 (его левый верхний угол)

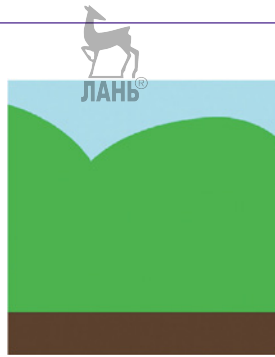


Рис. 9.3. Фрагмент рисунка 9.1 (его правый нижний угол)



Рис. 9.4. Фрагмент рисунка 9.1 (его левый нижний угол)

Например, на рисунке 9.2 представлен фрагмент рисунка 9.1 — его левый верхний угол. Модель машинного обучения не распознает эту картинку как изображение дерева, а мы поймем, что если на большой картинке и есть дерево, то оно не в левом верхнем углу.

Также мы можем попробовать распознать дерево в правом нижнем углу рисунка 9.1, он показан на рисунке 9.3. Модель машинного обучения снова не распознает эту картинку как изображение дерева, а мы поймем, что если на большой картинке и есть дерево, то оно не в левом верхнем углу и не в правом нижнем.

И так мы будем продолжать действовать до тех пор, пока не дойдем до левого нижнего угла рисунка 9.1, который показан на рисунке 9.4. Когда мы поймем, что модель машинного обучения с высокой степенью надежности распознала изображение дерева, мы будем знать, где именно на большой картинке находится дерево.

Чтобы было проще, представьте себе это так: вы разбиваете изображение на плитки и тестируете каждую плитку отдельно. В этой главе вы сами увидите, как работает этот метод, когда будете обучать модель машинного обучения находить определенные объекты на случайно сгенерированных Сценах.

Выполнение проекта

В этой главе вы создадите проект Scratch, который сначала будет генерировать случайный фон, а затем случайным образом добавлять на Сцену десятки разных спрайтов. Одним из этих спрайтов будет утка (рис. 9.5), и у модели машинного обучения будет задача найти эту утку. При этом найти ее нужно будет честно, т. е. именно распознать изображение утки, а не использовать координаты ее спрайта.



Рис. 9.5. Основная задача этого проекта — найти утку среди других объектов на большой картинке

Обучение модели

1. Создайте новый проект машинного обучения и назовите его «В поисках утки». В качестве распознаваемого типа данных выберите «изображения».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 9.6.

3. Нажмите на кнопку «Добавить новую метку» (рис. 9.7) для создания первой коллекции, назовите ее «Duck» (в переводе с англ.: утка).

4. Нажмите на кнопку «Добавить новую метку» снова и создайте коллекцию с именем «Not the Duck» (англ.: не утка), как показано на рисунке 9.8 (символы «нижнее подчеркивание» между словами в названии коллекции будут добавлены автоматически).

Эта коллекция будет использоваться для сбора «отрицательных» обучающих примеров, т. е. примеров того, что вы не хотите, чтобы компьютер научился распознавать.

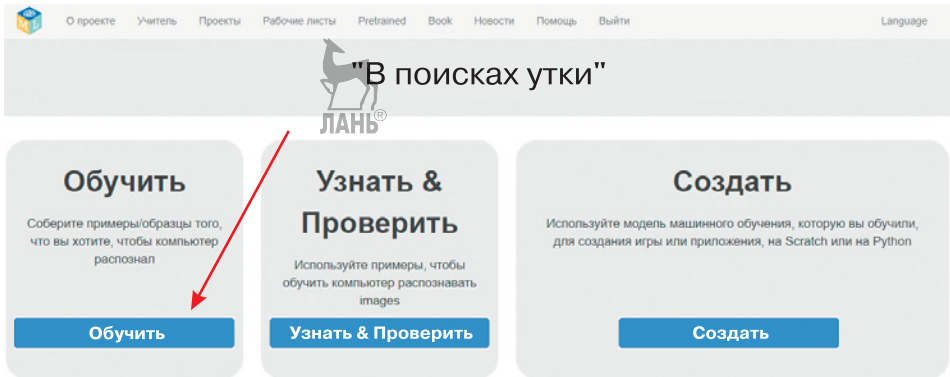


Рис. 9.6. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

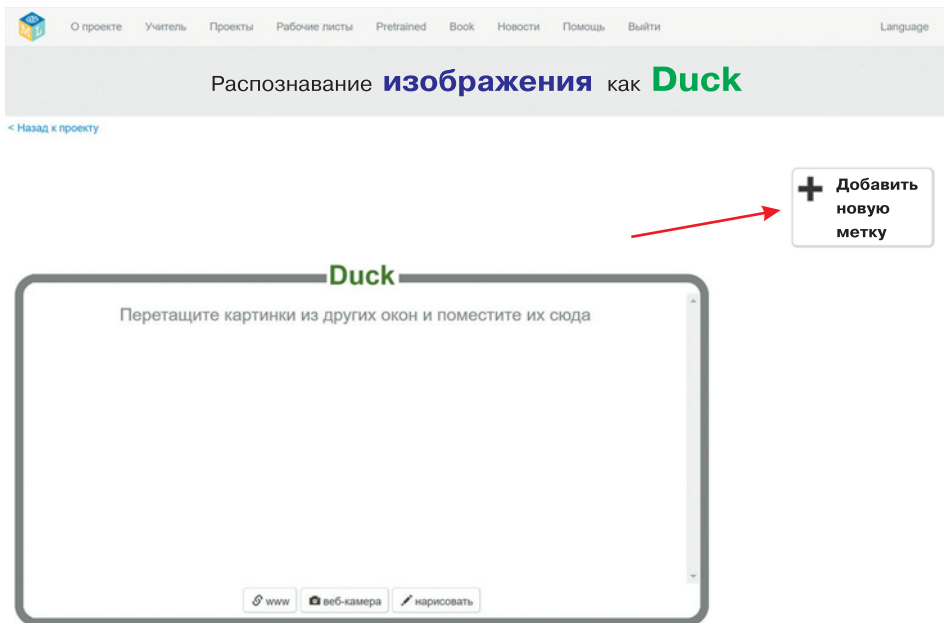


Рис. 9.7. Создание коллекции примеров изображений уток

5. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

6. Нажмите на кнопку «Создать», чтобы перейти к следующему этапу проекта.

7. Нажмите на кнопку «Scratch 3», как показано на рисунке 9.9.



Рис. 9.8. Создание коллекции примеров изображений, на которых может быть все что угодно, кроме уток

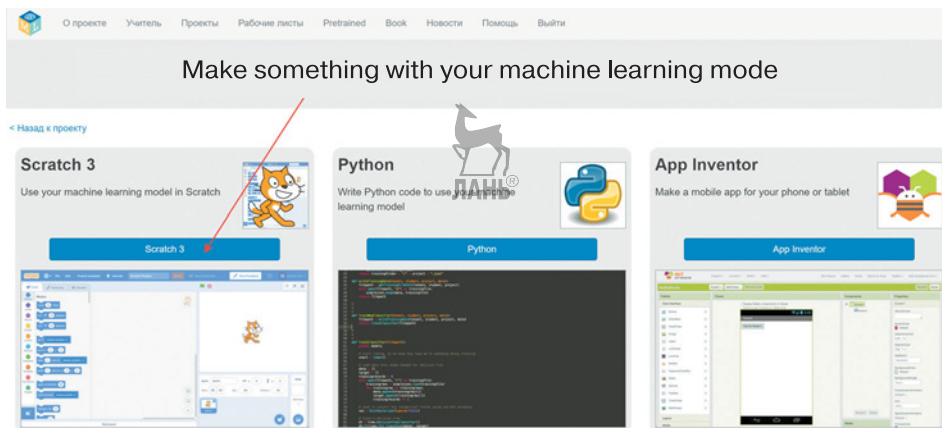


Рис. 9.9. Нажмите на кнопку «Scratch 3» для продолжения работы над проектом

Вы увидите предупреждение о том, что у вас еще нет модели машинного обучения. Это нормально, так как в этом проекте для сбора обучающих примеров вы будете использовать Scratch.

8. Нажмите на кнопку «Перейти к Scratch», как показано на рисунке 9.10.

9. В Главном меню выберите пункт **Шаблоны проектов**. В появившемся списке шаблонов выберите проект с именем «Find the Duck» (в переводе с англ.: найди утку).

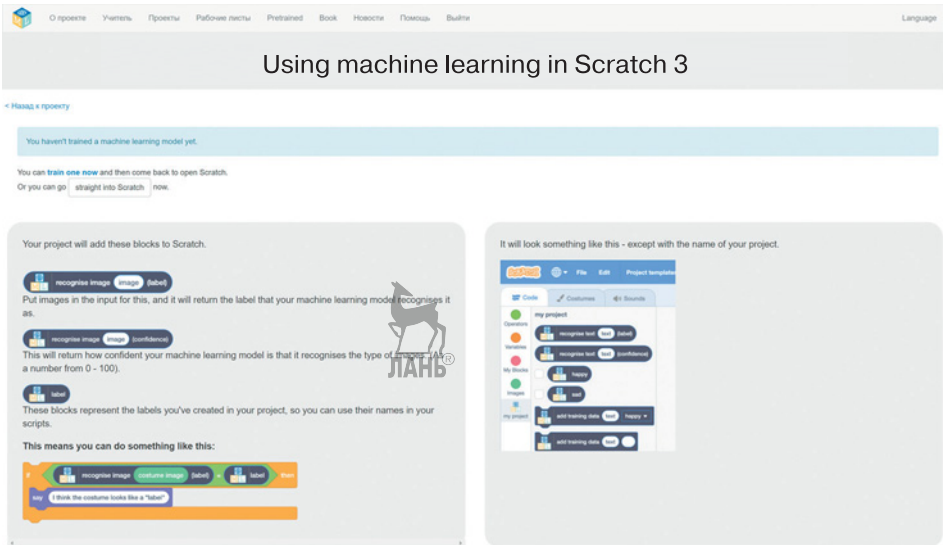


Рис. 9.10. Откройте Scratch без создания модели машинного обучения, нажав на кнопку «Перейти к Scratch»

В этом проекте на Сцене расположены 12 спрайтов. Сама Сцена представляет собой «сетку» размером 3×4 . Когда вы впервые запускаете проект, спрайты могут быть не видны, но они названы так, как показано на рисунке 9.11.

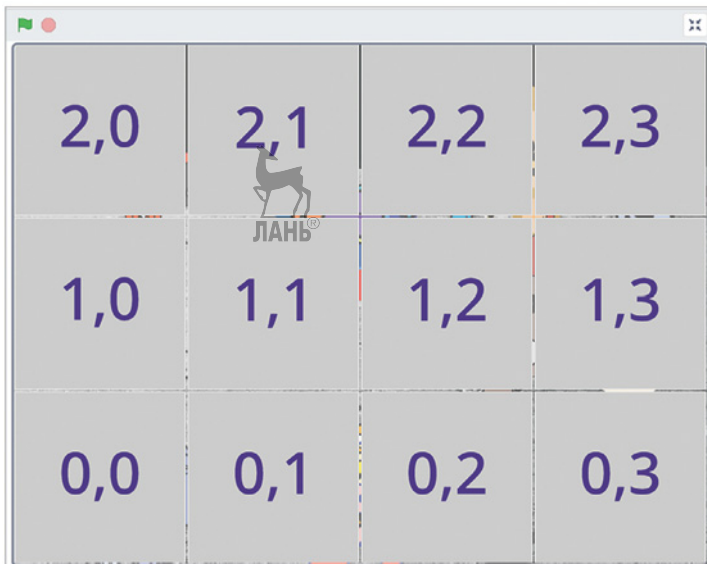


Рис. 9.11. Спрайты в шаблоне проекта «Find the Duck»

10. В списке спрайтов в правом нижнем углу экрана выберите спрайт с именем «0,0» и нажмите на него. Затем в левом верхнем углу Области кода найдите блоки «store training data example of the duck» (англ.: сохранить обучающий пример утки) и «store training data example of NOT the duck» (англ.: сохранить обучающий пример НЕ утки) (они расположены под большим желтым комментарием сверху «TRAINING»), как показано на рисунке 9.12.

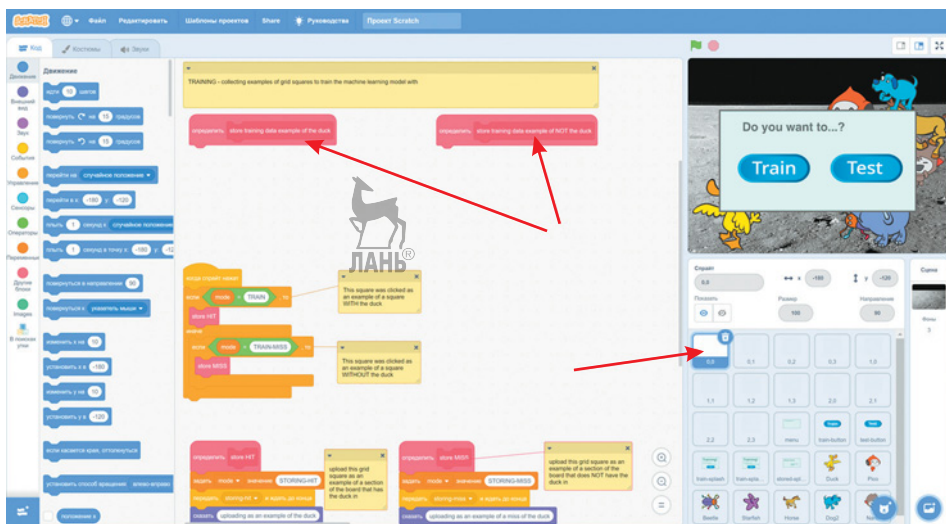


Рис. 9.12. Найдите фрагменты скрипта для спрайта 0,0

11. На Панели инструментов выберите категорию «Find the duck» и перетащите из нее в оба фрагмента скрипта блок «add training data» (англ.: добавить обучающие примеры). Затем из категории «Изображения» перетащите в оба фрагмента скрипта блок «backdrop image» и расположите его так, как показано на рисунке 9.13.

Настройте первый скрипт для добавления фона в коллекцию «Утка», а второй скрипт — для добавления фона в коллекцию «Не утка».

12. Повторите шаг 11 для всех двенадцати спрайтов (рис. 9.14). Как только вы это сделаете, сбор обучающих примеров начнется, на какой бы фрагмент экрана вы ни нажали.



Примечание

Если блок светится желтым, когда вы добавляете в него другие блоки, значит, вы случайно запустили ту часть скрипта, которая добавляет изображения в ваши коллекции обучающих примеров. Если это произойдет, вернитесь к этапу обучения и удалите эти изображения из коллекций.

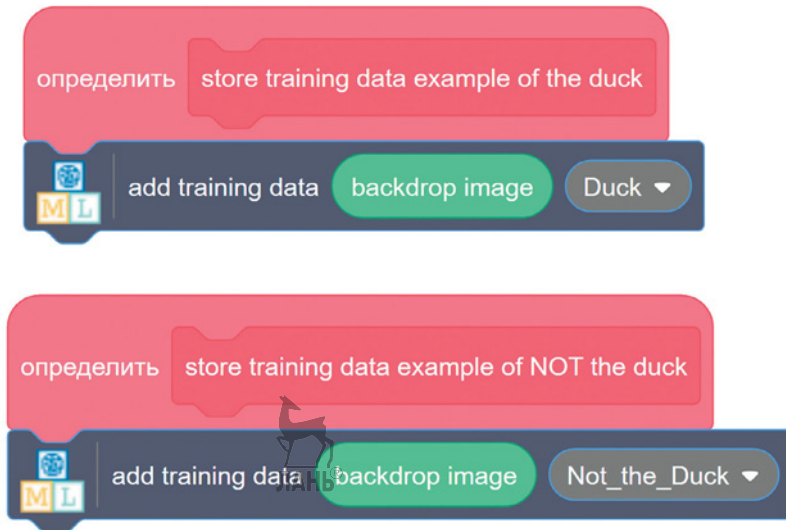


Рис. 9.13. Добавление обучающих примеров в две коллекции

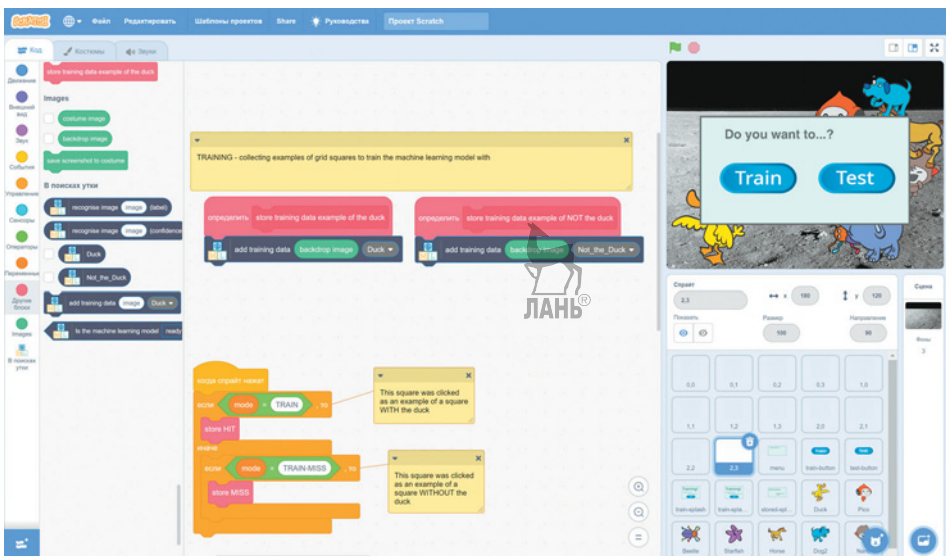


Рис. 9.14. Добавление блоков в скрипты для всех 12 спрайтов в проекте

13. Настало время сбора обучающих примеров! Нажмите на зеленый флажок, чтобы начать. Когда скрипт будет запущен, вы увидите сообщение «Do you want to... ?» (англ.: что вы хотите сделать?) и две кнопки — «Train» (англ.: обучить) и «Test» (англ.: протестировать). Нажмите на кнопку «Train».

Нажмите на кнопку «ОК», когда увидите сообщение с инструкцией (она предлагает вам нажать на кнопку «ОК» для начала обучения, а затем нажимать на спрайт с изображением утки каждый раз, когда вы его увидите). Тот фрагмент экрана, на который вы будете нажимать, будет добавляться в коллекцию обучающих примеров, как показано на рисунке 9.15.

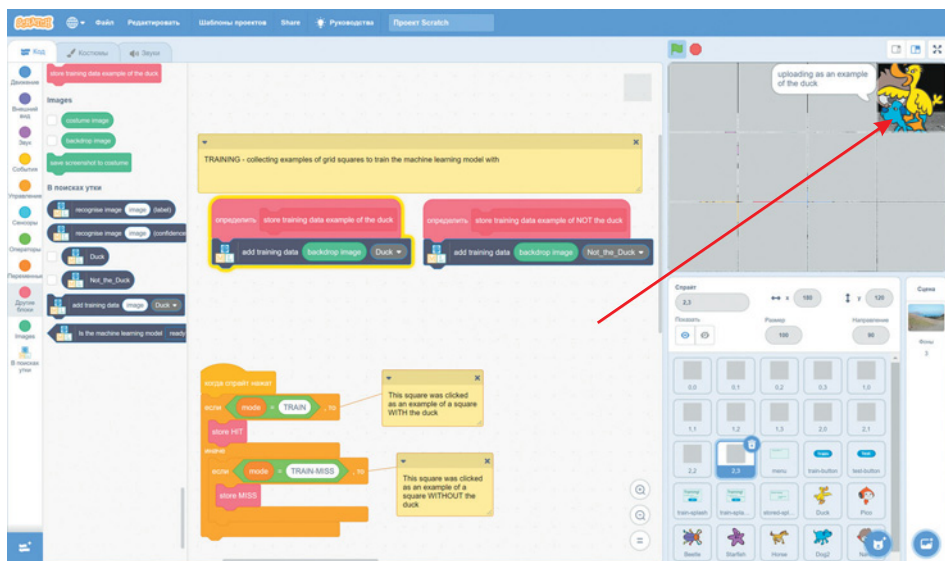


Рис. 9.15. Добавление обучающих примеров изображения утки



Примечание

Вся утка, скорее всего, не будет помещаться точно в одну из частей сетки, поэтому в коллекции обучающих примеров вы иногда будете видеть только ее фрагмент. Но совсем скоро вы убедитесь, что это все еще полезно и эффективно для обучения модели.

14. Нажмите на кнопку «ОК», когда вас попросят выбрать фрагмент экрана, на котором нет утки. Убедитесь, что вы нажали на тот фрагмент, на котором нет даже части утки.

15. Вернитесь на сайт проекта «Машинное обучение для детей», нажмите на ссылку «Назад к проекту», а затем перейдите к этапу обучения, нажав на кнопку «Обучить». Здесь вы сможете убедиться, что все работает хорошо. Вы должны увидеть в коллекциях все фрагменты экрана, на которые нажимали в проекте Scratch (рис. 9.16). Убедитесь, что каждый фрагмент находится в нужной коллекции.

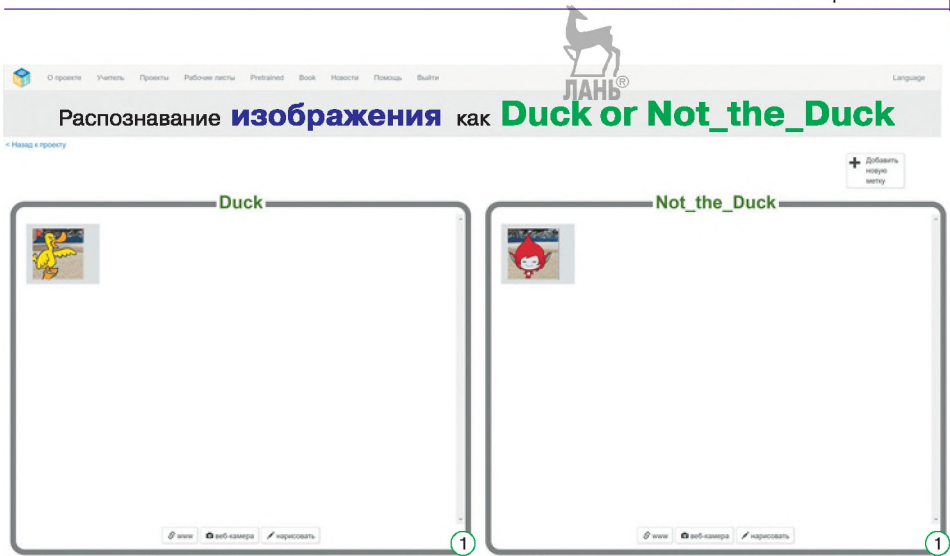


Рис. 9.16. Обучающие примеры должны появиться в нужных коллекциях, когда вы вернетесь к этапу обучения модели



Примечание

Если какой-либо фрагмент отсутствует, возможно, вы пропустили добавление блоков в одном из скриптов для спрайтов. Чтобы это проверить и исправить, вернитесь к проекту Scratch, и нажмите на каждый спрайт в списке спрайтов.

16. Вернитесь к проекту Scratch и повторяйте шаги 13–15 до тех пор, пока в каждой коллекции не соберется как минимум 10 обучающих примеров, как показано на рисунке 9.17.



Рис. 9.17. Обучающие примеры для проекта «В поисках утки»

Вот вам несколько подсказок для заполнения коллекции с изображениями не уток:

- Убедитесь, что вы не нажимаете на один и тот же объект каждый раз, когда вас просят найти не утку. Вы же не хотите, чтобы вторая коллекция стала обучающими примерами для распознавания одного конкретного персонажа (например, попугая)! Лучший способ создать хорошую коллекцию обучающих примеров для не утки — каждый раз выбирать разных персонажей.
- Попробуйте выбрать несколько фрагментов вообще без объектов на них. Пусть компьютер поймет, что пустые фрагменты (без животных) — тоже не утка.

17. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы. Затем нажмите на кнопку «Узнать и проверить», чтобы перейти к следующему этапу проекта. В нем нажмите на кнопку «Обучить модель машинного обучения», как показано на рисунке 9.18. Дождитесь, пока обучение модели завершится. Это может занять несколько минут.

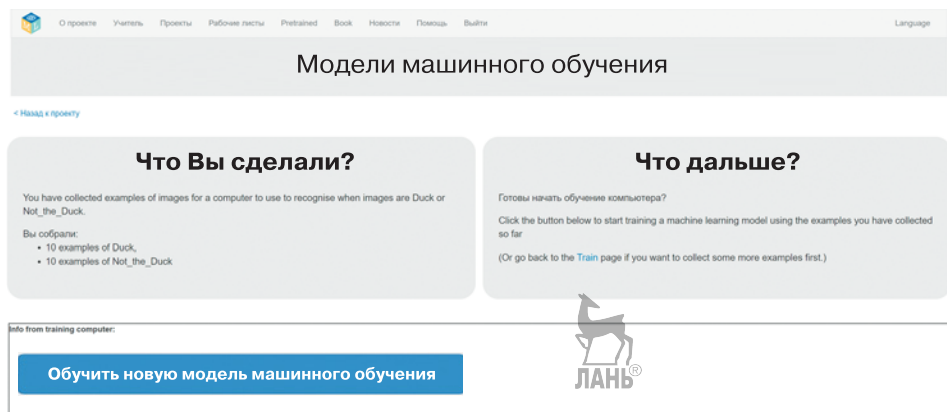


Рис. 9.18. Обучите модель машинного обучения на примерах, которые вы собрали в проекте Scratch

Подготовка проекта

Теперь вам нужно доработать проект Scratch, чтобы все скрипты можно было протестировать.

1. В списке спрайтов в правом нижнем углу экрана выберите спрайт с именем «0,0». Затем в Области кода найдите фрагмент скрипта «когда я получу test-0,0» (рис. 9.19). Он должен располагаться правее тех фрагментов скрипта, над которыми вы работали ранее.

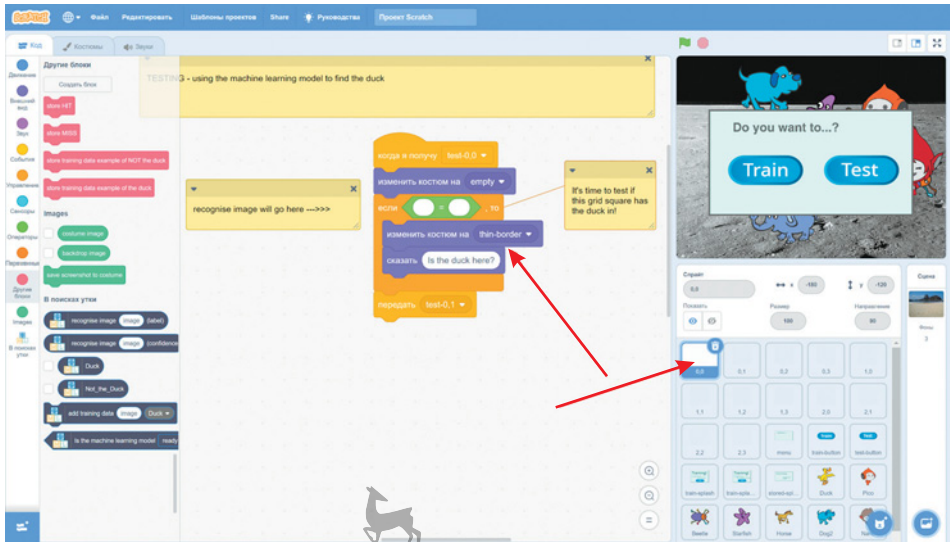


Рис. 9.19. Найдите скрипт, отвечающий за тестирование спрайта с именем «0,0»

2. В найденный фрагмент скрипта «когда я получу test-0,0» добавьте блок «recognise image (label)» (англ.: распознать изображение по имени) из категории с именем проекта машинного обучения «В поисках утки» и измените его так, как показано на рисунке 9.20.

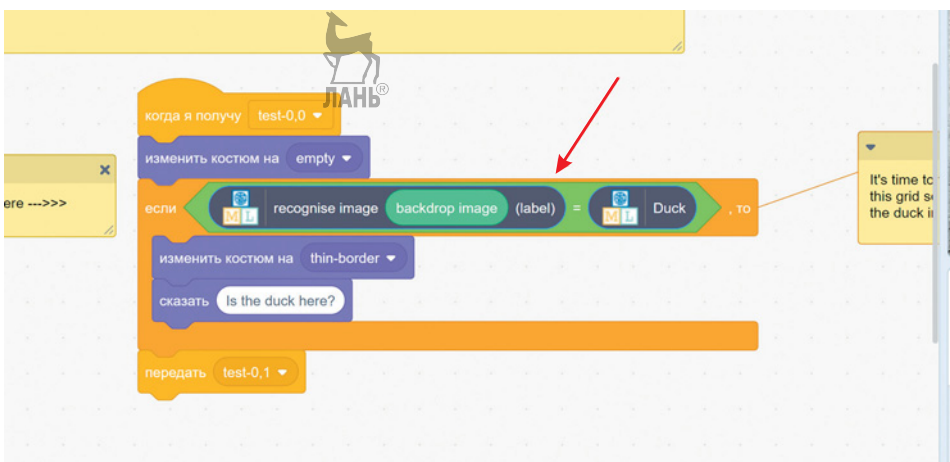


Рис. 9.20. Измените скрипт так, чтобы он использовал для тестирования вашу модель машинного обучения

Этот скрипт будет использовать вашу модель машинного обучения, чтобы определить, действительно ли левый нижний фрагмент экрана содержит изображение утки, а затем, если найдет, отобразит сообщение «Is the duck here?» (англ.: утка находится здесь?).

3. Повторите шаг 2 для всех 12 спрайтов (рис. 9.21). Как только вы это сделаете, ваша модель машинного обучения сможет искать утку на всех фрагментах экрана.

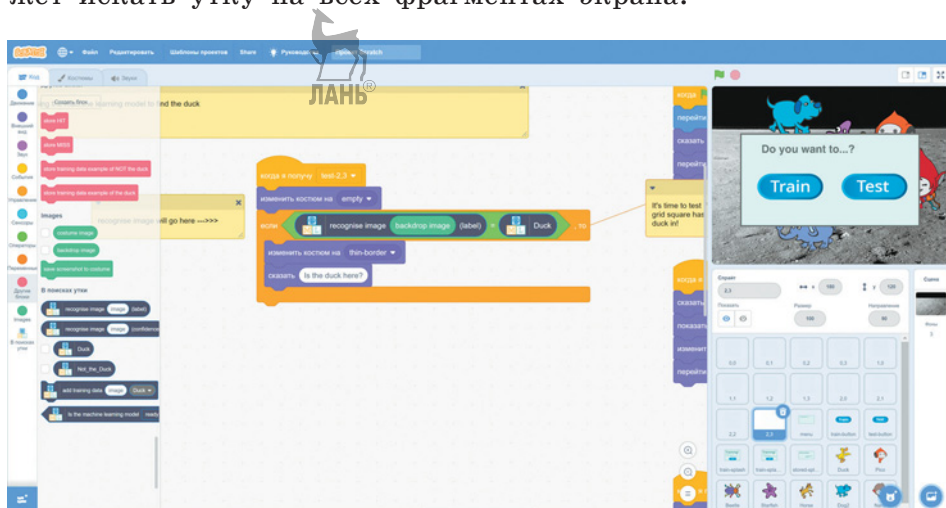


Рис. 9.21. Измените скрипт так, чтобы он использовал для тестирования вашу модель машинного обучения для всех 12 спрайтов

Тестирование модели

Настало время протестировать вашу модель машинного обучения и проверить, насколько хорошо она работает!

1. Нажмите на зеленый флажок, а затем на кнопку «Test» (англ.: протестировать).

Ваша модель машинного обучения попытается распознать утку на каждом из фрагментов экрана и подсветит фрагмент, в котором утка будет найдена (рис. 9.22).

2. Запустите проект несколько раз и посмотрите, как часто ваша модель машинного обучения дает правильный ответ. На самом деле поиск небольшого изображения на большой сцене, полной персонажей (имея всего 10 обучающих примеров), — это достаточно трудная работа, поэтому, если ваша модель несколько раз ошибется, — это нормально.



Рис. 9.22. Тестирование модели машинного обучения

3. Добавьте еще по 10 обучающих примеров в каждую из коллекций. Нажмите на зеленый флажок, а затем на кнопку «Train», как вы уже делали ранее. Чтобы посмотреть, что получается, вы всегда можете вернуться к проекту на этапе обучения (рис. 9.23).

Нажмите на ссылку «Назад к проекту», а затем на кнопку «Узнать и проверить», чтобы обучить модель машинного обучения на обновленных коллекциях обучающих примеров.

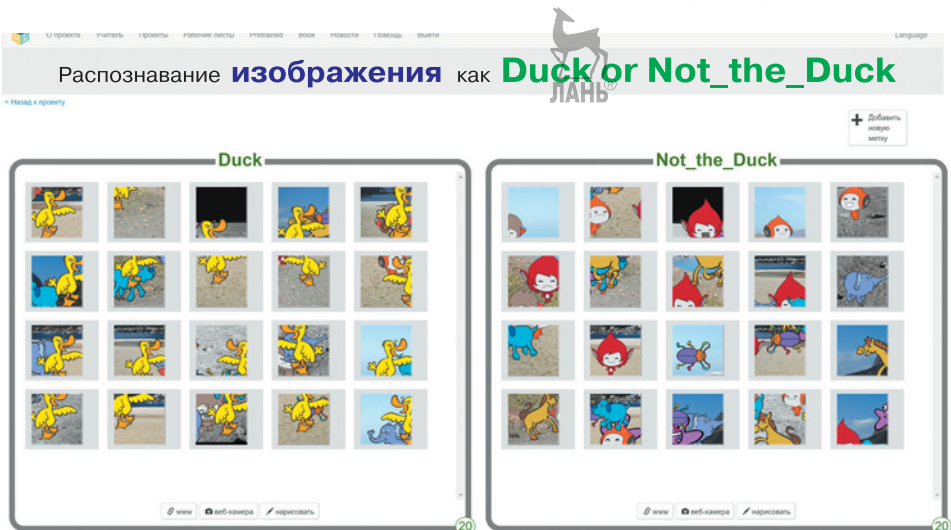


Рис. 9.23. Моя модель машинного обучения использует по 20 обучающих примеров в каждой из коллекций

4. Снова вернитесь к проекту Scratch и протестируйте обновленную модель машинного обучения. Стала ли она работать лучше?

Обзор и улучшение проекта

Как определить, насколько хорошо работает ваша модель машинного обучения?

В главе 8 вы узнали о нескольких способах оценки качества моделей машинного обучения. Для этого нужно было подсчитать, сколько раз модель дала верные ответы и сколько раз ошибалась. Чтобы составить матрицу ошибок, вы должны поделить ответы модели на 4 группы:

- **Истинно положительные.** Фрагмент экрана, который модель машинного обучения распознала как утку, на самом деле содержал утку.
- **Ложноположительные.** Фрагмент экрана, который модель машинного обучения распознала как утку, не содержал утку.
- **Истинно отрицательные.** Фрагмент экрана, который модель машинного обучения распознала как что-то другое, на самом деле содержал утку.
- **Ложноотрицательные.** Фрагмент экрана, который модель машинного обучения распознала как что-то другое, не содержал утку.

Вы можете использовать эти данные, чтобы составить матрицу ошибок и выявить долю правильных ответов, точность и полноту своей модели машинного обучения.



Примечание

Если вы не помните, как рассчитать эти показатели или как построить матрицу ошибок, вернитесь к разделу «Обзор и улучшение проекта» главы 8.

Взгляните на тестовый пример на рисунке 9.24.

Вы можете видеть, что на самом деле утка расположена сразу в четырех фрагментах экрана в правом нижнем углу. Но модель машинного обучения распознала утку только в двух фрагментах, а остальные два пропустила. Поэтому моя матрица ошибок выглядит так:

Истинно положительные 2	Ложноположительные 0
Ложноотрицательные 2	Истинно отрицательные 8

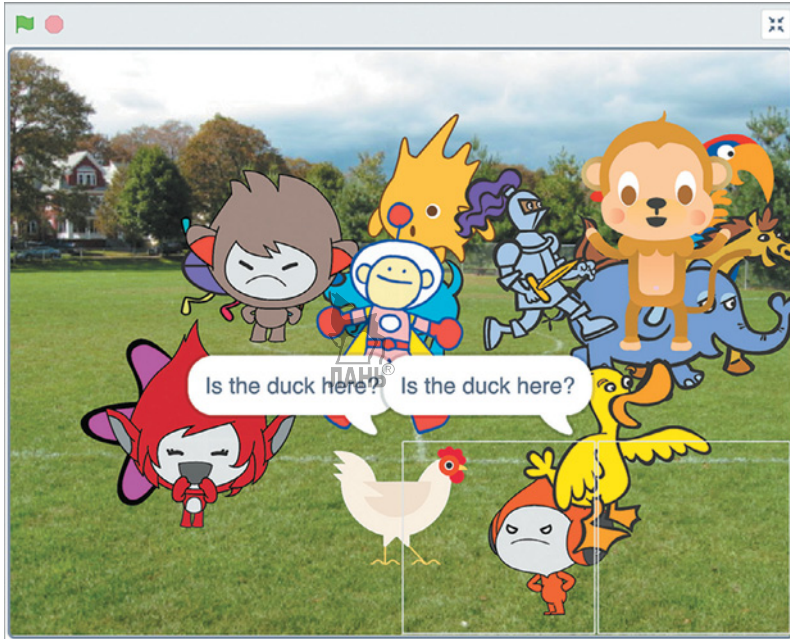


Рис. 9.24. Тестовый пример, на котором модель машинного обучения распознала утку на двух фрагментах экрана в правом нижнем углу

Моя матрица ошибок дает мне следующую информацию:

точность (precision) — 100% (каждый раз, когда моя модель думала, что распознала утку, там была утка);

полнота (recall) — 50% (моя модель распознала только половину фрагментов экрана, на которых на самом деле была утка);

показатель достоверности (ассигасу) — 83% (моя модель дала 10 верных ответов из 12).

Для того чтобы этим цифрам можно было доверять, вам понадобится гораздо больше данных, возможно, даже несколько разных фонов.

Я провел тестирование пять раз, и вот что у меня получилось:

Истинно положительные 9	ЛАНЬ®	Ложноположительные 0
Ложноотрицательные 6		Истинно отрицательные 45

Тогда получаем:

точность (precision) — 100%;

полнота (recall) — 60%;

достоверность (ассигасу) — 90%.

Эти числа дают нам больше информации для осмысления качества модели машинного обучения.

Моя модель, обученная на небольшом количестве примеров, кажется достаточно точной (когда она распознает утку, она всегда оказывается права). Тем не менее она не всегда распознает фрагмент утки как утку.

Полная модель, которая иногда что-то упускает, называется *предпочитающей точность полноте*. Такой подход хорош для проектов, где лучше не распознать объект, чем распознать его неправильно.

Для проектов, в которых важнее ничего не упустить и в которых допустимо определенное количество неверных распознаваний, вы должны стремиться обучать модели машинного обучения таким образом, чтобы они отдавали предпочтение полноте, нежели точности.

А как работает ваш проект?

Применение сложных систем распознавания изображений в реальных проектах

Возможно, вы уже обучали подобную модель машинного обучения распознаванию изображений. Вас когда-нибудь просили при переходе на веб-сайт подтвердить, что вы человек, выбором фрагментов изображений уличных знаков, как показано на рисунке 9.25? Или велосипедов? Или такси?

Думаю, вы уже понимаете, что такое приложение для распознавания изображений, известное как «CAPTCHA», может стать отличным инструментом для сбора большого количества обучающих примеров для системы распознавания изображений, которая может находить разные объекты на улице. Как вы думаете, будет ли это полезно при разработке, например, беспилотных автомобилей?

Основная идея, заложенная в сложных системах распознавания, описана во введении к этой главе. Если мы хотим найти что-то маленькое на большом изображении, мы разбиваем изображение на более мелкие фрагменты и тестируем каждый фрагмент по отдельности с помощью модели машинного обучения, которая умеет распознавать изображения этого объекта.

Возможно, вам было не очень просто сразу разобраться с этим методом, изучая его самостоятельно. Например, одной из самых больших проблем является поиск ответа на вопрос, какой размер фрагментов использовать. Помните пример поиска дерева на рисунке 9.1?

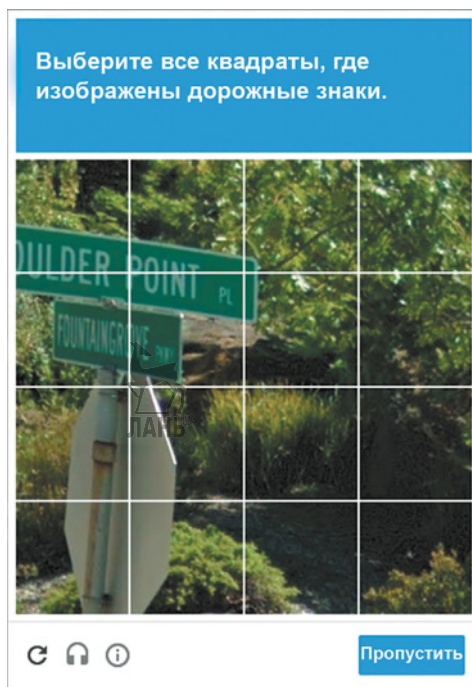


Рис. 9.25. Помощь в обучении моделей машинного обучения

Если вы сделаете фрагменты слишком маленькими, вы сможете увидеть только небольшую часть объекта, который пытаетесь найти, и так никогда и не распознать его. Например, ваша модель машинного обучения может не распознать фрагмент на рисунке 9.26 как дерево.



Рис. 9.26. Фрагмент изображения, который содержит только небольшую часть дерева



Рис. 9.27. Фрагмент изображения слишком большой для того, чтобы на нем можно было распознать дерево

С другой стороны, если вы сделаете фрагменты слишком большими (как на рисунке 9.27), вы можете столкнуться с обратной ситуацией, когда модель машинного обучения не сможет распознать дерево, потому что на ее обучающих примерах деревья были гораздо больше.

Если вы знаете вероятный размер объекта, который вы ищете на картинке, вы можете правильно оценить необходимый размер фрагментов для распознавания. Некоторые системы даже просят пользователей указать конкретный размер распознаваемых фрагментов.

Если ни одно из этих решений не подходит, вы можете попробовать поварьировать размерами фрагментов и в конечном счете остановиться на том размере, который приведет вашу модель к максимальной степени надежности, как показано на рисунке 9.28.

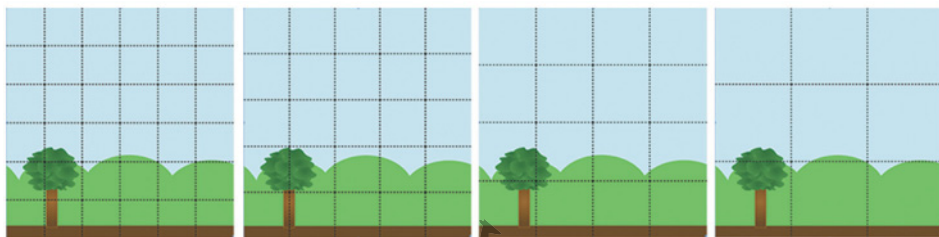


Рис. 9.28. Попробуйте поэкспериментировать с размерами фрагментов, если заранее не знаете, какой размер подойдет лучше всего

И даже если вы выберете правильный размер фрагментов, объект, который вы ищете, не всегда будет точно размещаться в середине фрагмента (как вы, вероятно, заметили в случае с уткой).

А еще, чтобы увеличить шанс найти фрагмент с нужным объектом посередине, вам также стоит попробовать разные начальные координаты.

Системы, использующие комбинацию этих методов, могут быть очень эффективными. Например, в 2015 году во время чрезвычайного положения, вызванного засухой в Калифорнии, модель машинного обучения использовалась для поиска газонов, бассейнов и других объектов, влияющих на дефицит воды.

Разделение спутниковых изображений всего штата Калифорнии на фрагменты, точно так же как вы сделали в своем проекте, означало, что каждый фрагмент можно было классифицировать по отдельности. Основное отличие от вашего проекта заключалось в том, что калифорнийская модель машинного обучения была обучена распознавать не один, а несколько разных

объектов, влияющих на водорасход. (В главе 3 вы видели, как можно обучить модель машинного обучения распознавать изображения различных объектов.) Сочетание распознавания изображений с картой позволило правительству Калифорнии быстро ознакомиться с ситуацией расходования воды в штате.

Калифорния — огромный штат, и для проведения такого исследования или опроса вручную потребовалось бы много времени. Машинное обучение оказалось быстрым и эффективным инструментом, а в чрезвычайных ситуациях скорость и эффективность критически важны.

Методы распознавания изображений моделями машинного обучения также регулярно используются в бизнесе. Например, дроны могут делать фотографии высокого разрешения, пролетая над зданиями, крышами, мостами, солнечными панелями, трубами и многими другими объектами. Затем эти фотографии нарезаются на фрагменты и распознаются моделью машинного обучения. Она может распознать, например, признаки повреждений или плохого обслуживания и ремонта. Автоматизированные системы распознавания изображений, основанные на тех же принципах, что и проект этой главы, используются в различных областях, таких как строительство (для осмотра мостов и зданий), сельское хозяйство (для распознавания здоровых или больных растений и сельскохозяйственных культур) или даже общественная безопасность (например, в Австралии, где машинное обучение используется в дронах-спасателях, которые могут распознавать акул с воздуха).

Что вы узнали

В этой главе вы обучили модель машинного обучения распознавать объекты, являющиеся частью более крупной картинки. Это самый сложный проект из тех, что вы делали до сих пор, но, надеюсь, теперь вы хорошо понимаете как строятся сложные системы распознавания изображений. Вы узнали о некоторых трудностях обучения таких систем, например о том, как разбить сложную задачу на более простые задачи (например, путем выбора правильного размера плитки), и получили несколько советов по их решению. Вы также познакомились с некоторыми реальными проектами таких сложных систем машинного обучения и узнали, в каких сферах они применяются.

В следующей главе вы рассмотрите еще один распространенный способ применения машинного обучения.



ГЛАВА 10

Умные помощники



В этой главе вы познакомитесь с возможностями повседневного использования машинного обучения, а именно — с умными помощниками. Возможно, вы уже слышали о Siri, Alexa, Google Home или об Алисе — это умные помощники, которые могут выполнять за вас много интересных вещей. Например, они могут установить будильник или включить музыку, если вы их об этом попросите. Здорово, правда?

На самом деле умные помощники — это тоже системы машинного обучения, только они обучены распознавать речь. Из предыдущего проекта вы уже знаете, как можно научить компьютер распознавать рукописный текст. Теперь же вы научите его распознавать ваши указания. И если компьютер сможет понять, что вы имели в виду, он сможет для вас выполнять определенные действия.



Вам нужно будет собрать большое количество примеров команд, которые вы хотите, чтобы компьютер умел выполнять. Эти примеры вы, как всегда, будете использовать для обучения модели машинного обучения. Затем вы напишете программу, которая будет использовать принцип **классификации намерений** (англ.: *intent classification*). Она будет рассматривать полученный текст как намерение, т. е. указание что-то сделать.

Вы еще не встречали термин «классификация намерений», но если вы выполнили предыдущие проекты из этой книги — вы уже с ним отчасти знакомы. Поздравляем! Например, вы уже научили компьютер классифицировать сообщения как комплименты или оскорбления, а заголовки газет — как таблоиды или новостные газеты. Еще вы познакомили компьютер со стилями письма, и теперь, когда вы даете ему какой-то текст, компьютер попытается классифицировать этот текст, т. е. определить, к какой из категорий этот текст должен относиться. Что же касается слова «намерение», в данном случае речь идет о том, чтобы научить компьютер не просто распознать текст и отнести его к какой-то категории,

а попытаться распознать в этом тексте указание что-то сделать. Это и означает «классифицировать намерение».

Классификация намерений полезна для создания компьютерных систем, с которыми мы можем взаимодействовать естественным образом (например, говорить). Это работает так: если вы говорите: «Включи свет», это означает, что вы *намереваетесь* включить свет. Для подобного описания есть название — *интерфейс естественного языка*. Другими словами, для того чтобы включить свет, вместо того чтобы нажимать на выключатель (или даже на кнопку в приложении или на сайте), вы используете *естественный язык* — язык, который естественным образом развился у людей, а не предназначен для компьютеров. Именно с помощью естественного языка вы сообщаете компьютеру о своем намерении. Прямо как живому человеку.

Вы будете давать компьютеру примеры, а он, в свою очередь, будет пытаться выявить в них закономерности. Закономерностью может быть выбор вами определенных слов при формулировании команд, способы сочетания этих слов (наборы слов) или даже использование более коротких или более длинных команд.

В этой главе вы создадите собственного виртуального умного помощника, который сможет распознавать ваши команды и выполнять соответствующие действия. Пример работы проекта представлен на рисунке 10.1.

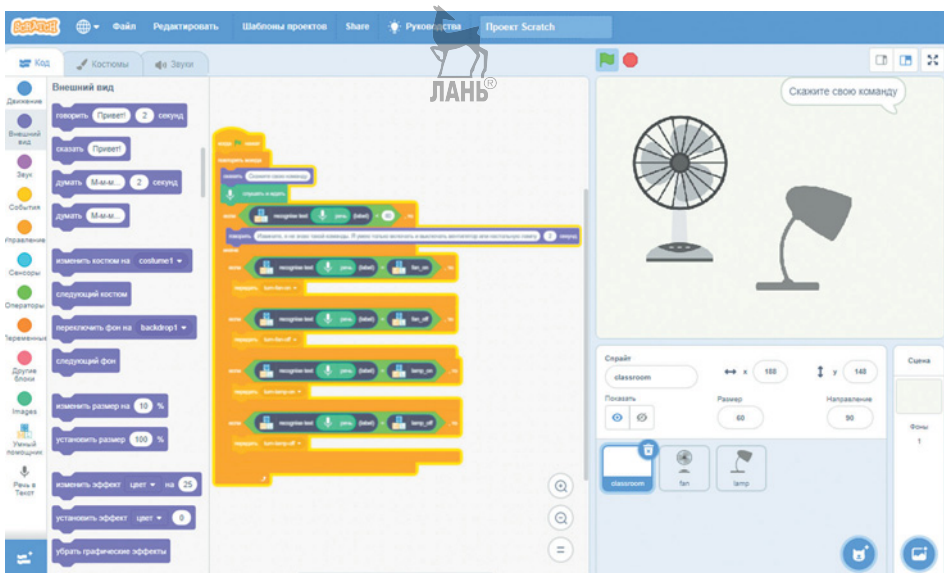


Рис. 10.1. Создание умного помощника с помощью Scratch

Звучит интересно, правда? Давайте же скорее приступим!

Выполнение проекта

Для начала вы создадите модель машинного обучения, которая сможет распознавать команды для включения и выключения двух устройств — настольной лампы и вентилятора.

Создание программы без использования машинного обучения

Когда вы выполняли проект из главы 7, вы убедились, что оценить все удобство машинного обучения можно, сначала выполнив проект без его использования. Впрочем, вы можете пропустить этот шаг, если хорошо понимаете разницу между программами, основанными на пошаговых инструкциях, и программами, использующими машинное обучение. Тогда вы можете сразу перейти к части проекта, где используется модель машинного обучения.

1. Зайдите в Scratch через сайт проекта <https://machinelearningforkids.co.uk/scratch3/>
2. В Главном меню выберите пункт **Шаблоны проектов** (рис. 10.2).
3. Выберите шаблон «Smart Classroom» (англ.: умный класс).

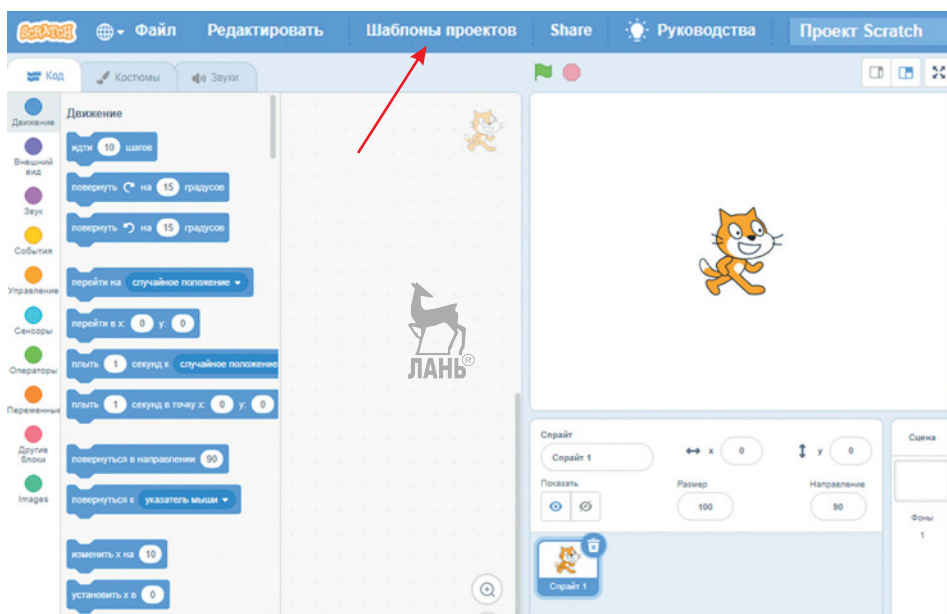


Рис. 10.2. Шаблоны проектов в Scratch содержат частичную реализацию проектов, чтобы сэкономить ваше время на начальном этапе

4. Создайте такой же скрипт, как на рисунке 10.3.

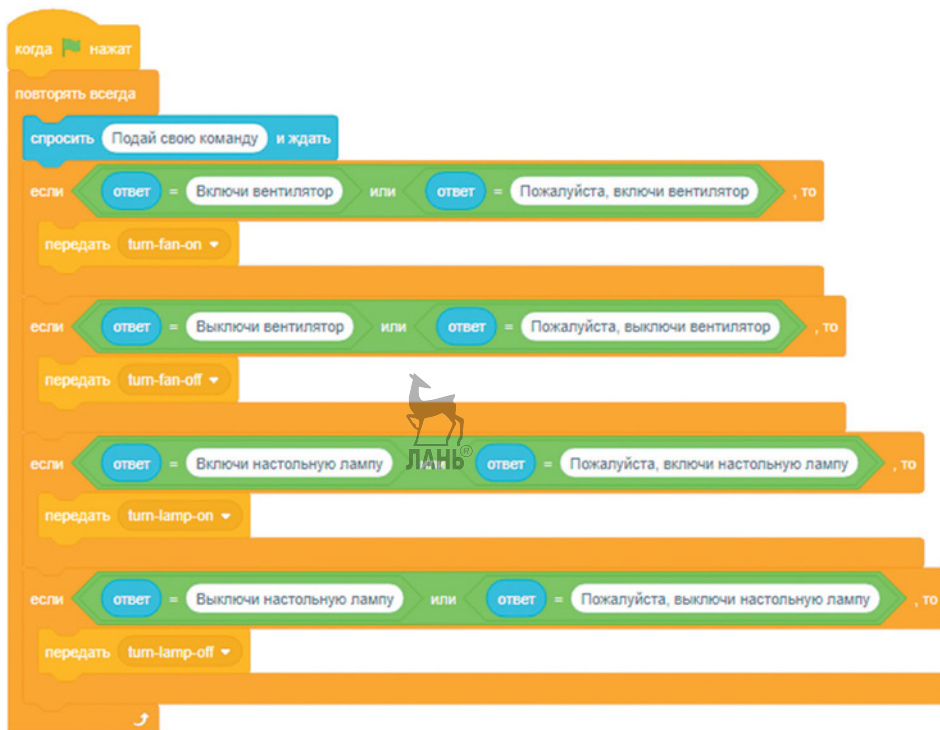


Рис. 10.3. Программирование пошаговых инструкций для умного помощника

Этот скрипт запрашивает у пользователя команду. Если пользователь введет «Включи вентилятор» или «Пожалуйста, включи вентилятор», «Выключи вентилятор» или «Пожалуйста, выключи вентилятор», «Включи настольную лампу» или «Пожалуйста, включи настольную лампу», «Выключи настольную лампу» или «Пожалуйста, выключи настольную лампу», Scratch воспроизведет соответствующую анимацию — анимацию включения или выключения настольной лампы или вентилятора. Попробуйте запустить проект и убедитесь в этом сами.

5. Нажмите на зеленый флажок. После запуска проекта введите команду «Включи вентилятор». Вы увидите, как лопасти вентилятора начали крутиться.

Но что будет, если вы напишете какую-то другую команду или допустите опечатку в слове? Что будет, если вы перефразируете команду? Например, не «Пожалуйста, включи вентилятор», а «Включи, пожалуйста, вентилятор». А что будет, если вы вообще не используете слово «вентилятор» в своей команде? Напри-

мер, вы могли бы написать: «Мне очень жарко, в этой комнате явно не хватает прохладного воздуха». Человек понял бы вас в любом из этих случаев, а этот скрипт нет. Все эти варианты для него — непонятные и неправильные.

Но что же делать? Возможно ли создать скрипт, который знал бы и учитывал все возможные фразы, означающие просьбу включить и выключить вентилятор или настольную лампу?

Вспомните самое начало главы 1, где я объяснял вам, что такое искусственный интеллект. Там же вы узнали, что машинное обучение не единственный способ создания систем искусственного интеллекта. Только что вы создали проект, в основе которого написание пошаговых инструкций, а не машинное обучение. Вы самостоятельно убедились, что этот способ не подходит для решения поставленной задачи. Именно так вы сможете быстрее и лучше понять, чем хорошо машинное обучение и почему оно на сегодняшний день используется в стольких проектах.

Обучение модели

1. Создайте новый проект машинного обучения* и назовите его «Умный помощник». В качестве распознаваемого типа данных выберите «текст».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.



2. Нажмите на кнопку «Обучить», как показано на рисунке 10.4.

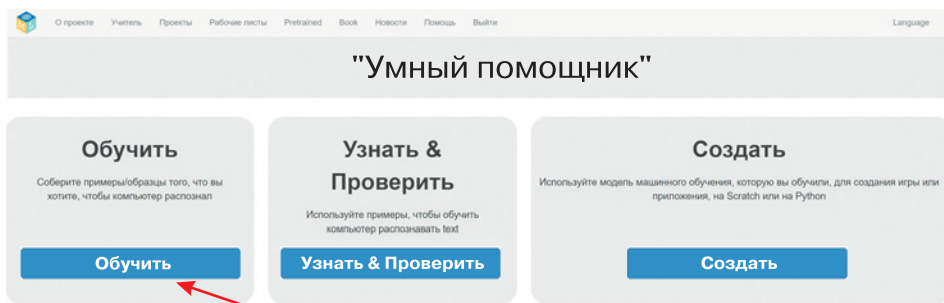


Рис. 10.4. Переход к первому этапу проекта машинного обучения — сбору обучающих примеров

* Создавайте проект в режиме гостя без входа в свой аккаунт. — Прим ред.

3. Нажмите на кнопку «Добавить новую метку» (рис. 10.5) и создайте первую коллекцию обучающих примеров. Назовите ее «fan on» (англ.: включить вентилятор). Повторите этот шаг еще 3 раза и создайте коллекции с именами «fan off» (англ.: выключить вентилятор), «lamp on» (англ.: включить настольную лампу) и «lamp off» (англ.: выключить настольную лампу). Символы «нижнее подчеркивание» между словами в названии коллекции будут добавлены автоматически.

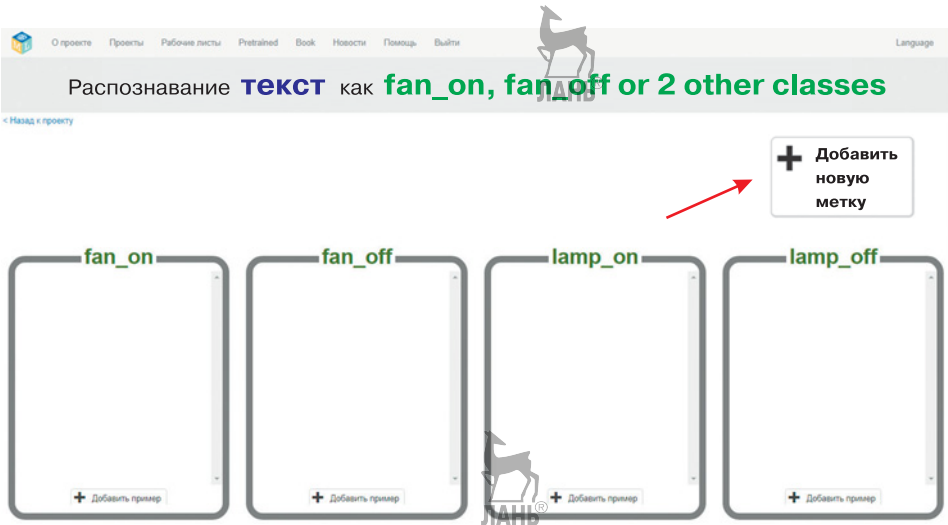


Рис. 10.5. Создание коллекций для четырех команд, которые компьютер должен будет научиться распознавать

4. Нажмите на кнопку «Добавить пример», чтобы добавить новый обучающий пример в коллекцию «fan_on». Напишите любую фразу, которую вы могли бы использовать в повседневной жизни, чтобы попросить кого-то включить вентилятор (рис. 10.6).

Фразы для обучающих примеров могут быть любыми.

Они могут быть короткими или длинными (например, «Включи вентилятор» или «Не мог бы ты включить вентилятор? Мне очень жарко»).

Они могут быть вежливыми или менее вежливыми (например, «Включи, пожалуйста, вентилятор» или «Срочно включи вентилятор»).

Они могут содержать слова «вентилятор» и «включить» или же не содержать их (например, «Включи, пожалуйста, вентилятор» или «Мне очень жарко, в этой комнате явно не хватает прохладного воздуха»).

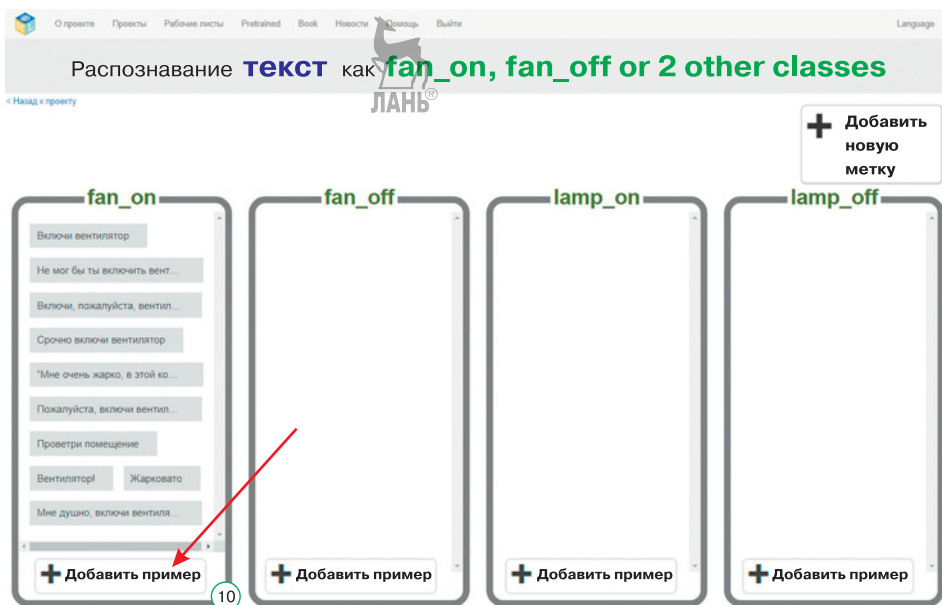


Рис. 10.6. Сбор обучающих примеров — фраз, которыми можно попросить кого-то включить вентилятор

Добавьте в эту коллекцию столько обучающих примеров, сколько фраз на тему включения вентилятора придет вам на ум. Но помните, что в каждой коллекции должно быть не меньше пяти обучающих примеров. В моем проекте я добавил шесть, и вы тоже можете их использовать, поэтому, надеюсь, у вас не возникнет с этим проблем.

5. Теперь нажмите на кнопку «Добавить пример», чтобы добавить новый обучающий пример в коллекцию «fan_off» (рис. 10.7). В эту коллекцию вам снова нужно будет добавить не менее пяти обучающих примеров — фраз, означающих просьбу выключить вентилятор. Как и на предыдущем шаге, добавляйте разнообразные фразы, а также попробуйте не использовать слова «вентилятор» и «выключить». Уверен, у вас получится!

6. Повторите шаги 4 и 5 для заполнения обучающими примерами двух оставшихся коллекций. Помните, в каждой коллекции должно оказаться не менее пяти примеров (рис. 10.8).

7. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

8. Нажмите на кнопку «Узнать и проверить», чтобы перейти к следующему этапу проекта.

Распознавание **ТЕКСТ** как **fan_on, fan_off or 2 other classes**[← Назад к проекту](#)

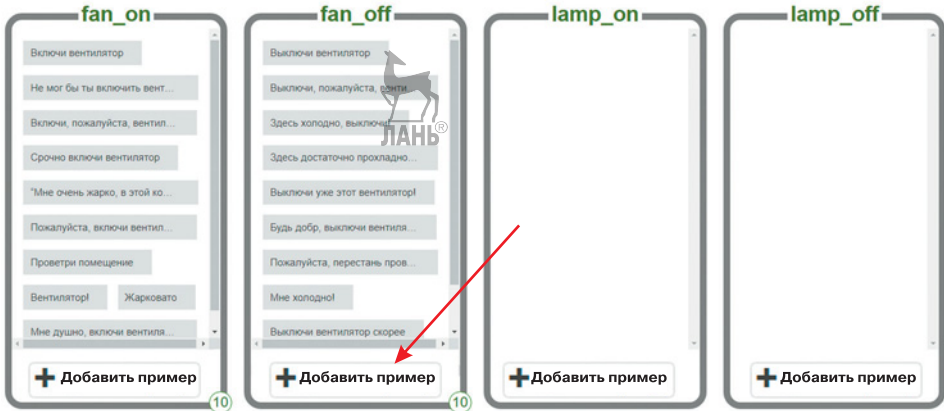
**Добавить
новую
метку**


Рис. 10.7. Сбор обучающих примеров для второй коллекции — фраз, которыми можно попросить кого-то выключить вентилятор

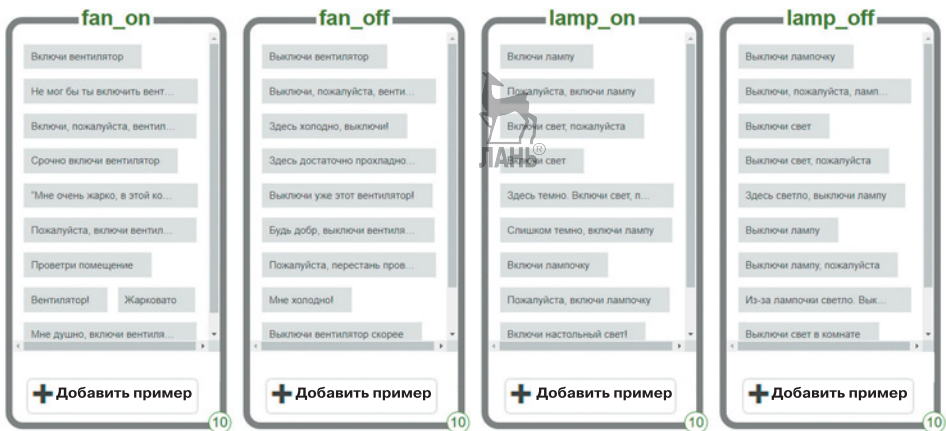


Рис. 10.8. Обучающие примеры для проекта «Умный помощник»

9. Нажмите на кнопку «Обучить новую модель машинного обучения», как показано на рисунке 10.9.

Компьютер будет использовать в качестве обучающих примеров те фразы, которые вы написали, и в итоге должен научиться распознавать содержащиеся в них команды. Это может занять несколько минут.

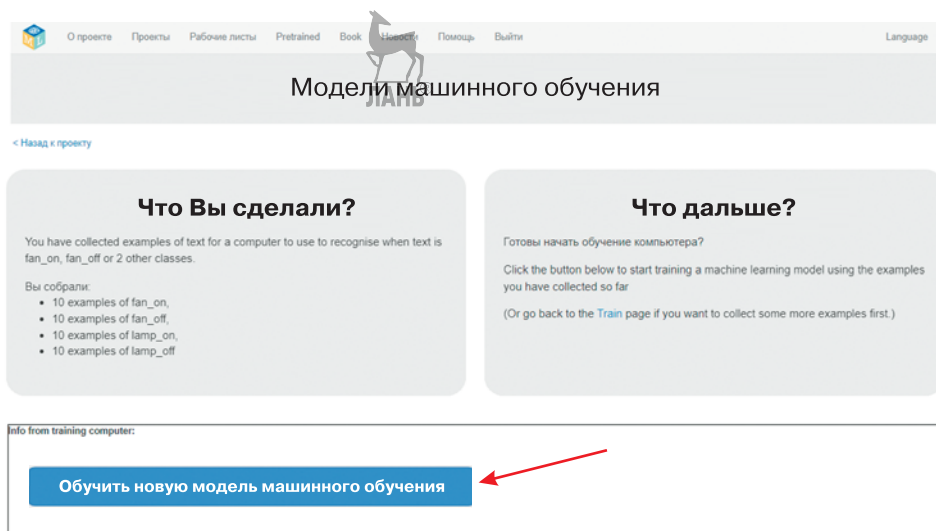


Рис. 10.9. Обучите модель машинного обучения для проекта «Умный помощник»

10. После того как обучение модели завершится, вам нужно будет протестировать ее и посмотреть, насколько хорошо она работает и распознает команды.

Для этого напишите любую команду в поле для ввода текста, как показано на рисунке 10.10.

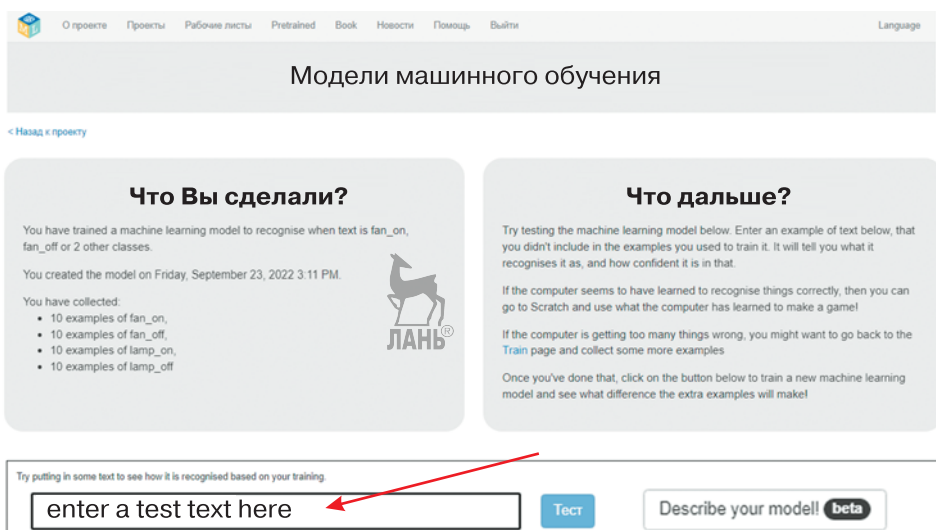


Рис. 10.10. Тестирование модели машинного обучения



Примечание

Очень важно, чтобы тестовые примеры не совпадали с обучающими примерами. Вам нужно понять, насколько хорошо ваша модель обучилась распознавать новые примеры, которые ей еще не попадались, а не насколько хорошо она научилась запоминать обучающие примеры.

Если модель допускает ошибки, вернитесь к этапу обучения и добавьте больше обучающих примеров в коллекции. При этом постарайтесь добавить побольше таких примеров, которые похожи на те, с которыми модель не справляется при тестировании. Вспомните, как учитель после контрольной работы подробно разбирает в первую очередь те задания, которые вызвали у учеников затруднения.

После того как вы соберете достаточное количество обучающих примеров, не забудьте перейти к этапу «Узнать и проверить», чтобы заново обучить свою модель. Теперь можно протестировать ее снова и посмотреть, стала ли она работать лучше.

Создание программы с использованием модели машинного обучения

Теперь у вас есть модель машинного обучения, которая может распознавать ваши команды. Значит, вы можете переделать проект Scratch и на этот раз использовать вашу модель машинного обучения вместо программирования пошаговых инструкций.

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

2. Нажмите на кнопку «Создать».

3. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch.

На Панели инструментов вы увидите новую категорию блоков, которая будет называться так же, как и ваш проект (рис. 10.11).

4. В Главном меню Scratch (в верхней части экрана) выберите пункт Шаблоны проектов. Так же как и в начале проекта этой главы, выберите шаблон «Smart Classroom» (англ.: умный класс).

5. Воспроизведите скрипт, который показан на рисунке 10.12.

Этот скрипт будет использовать вашу модель машинного обучения. Когда вы запустите проект и будете вводить команды, именно она будет их распознавать и выполнять нужные действия.

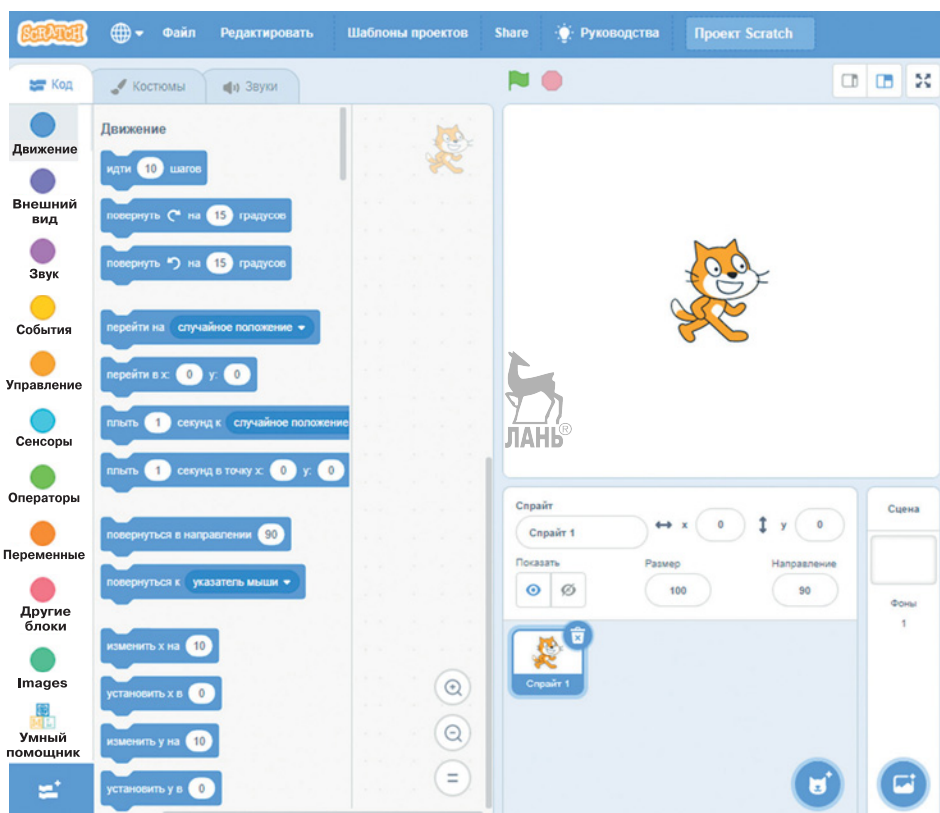


Рис. 10.11. На Панели инструментов появилась категория блоков вашего проекта машинного обучения

Тестирование модели

Настало время протестировать вашу модель машинного обучения и проверить, насколько хорошо она работает! Нажмите на зеленый флажок и по очереди введите несколько разных команд. При этом используйте самые разнообразные фразы — вы же хотите узнать, насколько хорошо будет справляться ваша модель? Теперь сравните, насколько лучше работает ваш умный помощник по сравнению с его предыдущей версией, которая использовала программирование пошаговых инструкций вместо машинного обучения.

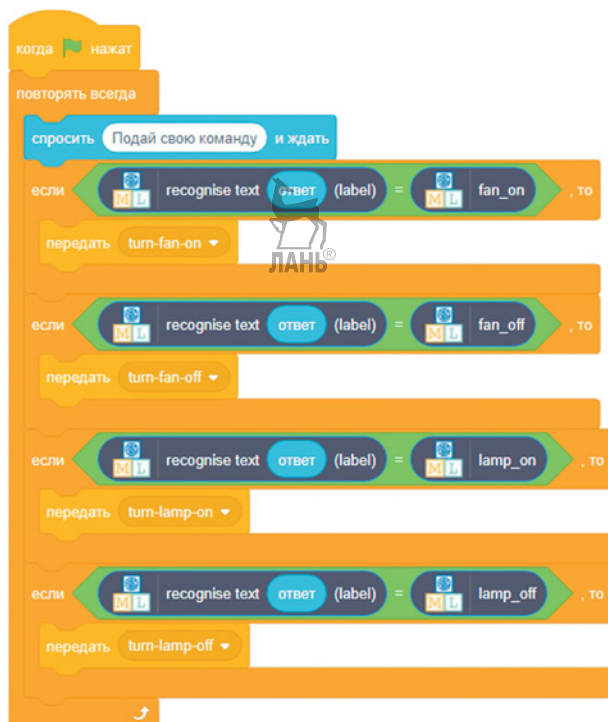


Рис. 10.12. Выполнение проекта «Умный помощник» вторым способом — с использованием модели машинного обучения

Обзор и улучшение проекта

Ура! Вы только что создали своего первого умного помощника, который может понимать и выполнять ваши команды. По сути, это упрощенная версия таких известных умных помощников, как Siri или Алиса. Теперь давайте посмотрим, что вы можете сделать, чтобы улучшить этот проект.

Использование показателя степени уверенности вашей модели машинного обучения

Вернитесь ко второму этапу проекта машинного обучения — этапу «Узнать и проверить». На странице этого этапа вы наверняка заметили количественную оценку — **степень уверенности** (confidence scores), когда тестировали свою модель. Вы уже знакомы с этим показателем из проекта главы 7. Компьютер отображает степень уверенности, чтобы показать, насколько он уверен в полученном результате распознавания тестового примера.

Теперь попробуйте ввести в поле для ввода тестовых примеров любую фразу, которая не будет похожа ни на одну из четырех команд, которые должна распознавать ваша модель.

Например, вы можете написать «Какой город — столица России?», как показано на рисунке 10.13.

Моя модель распознала эту фразу как команду включить настольную лампу, но при этом показала степень уверенности 0%. Именно так модель пытается показать, что не смогла распознать полученный тестовый пример.

Фраза «Какой город — столица России?» не похожа ни на один из обучающих примеров, которые я предоставлял своей модели машинного обучения. Этот вопрос не содержит ни одну из закономерностей, которые компьютер научился распознавать. Это означает, что модель машинного обучения не смогла распознать в этом вопросе ни одну из четырех команд для умного помощника.

Ваша модель машинного обучения может показать степень уверенности и не 0%, но это все еще будет не очень большое число. Если же это не так и процент достоверности будет высоким, вернитесь к этапу обучения и соберите больше обучающих примеров.

Продолжайте экспериментировать — тестируйте свою модель фразами, не имеющими отношения ни к вентилятору, ни к настольной лампе. Сравните полученные значения степени уверенности с теми случаями, когда вы вводите настоящие команды по

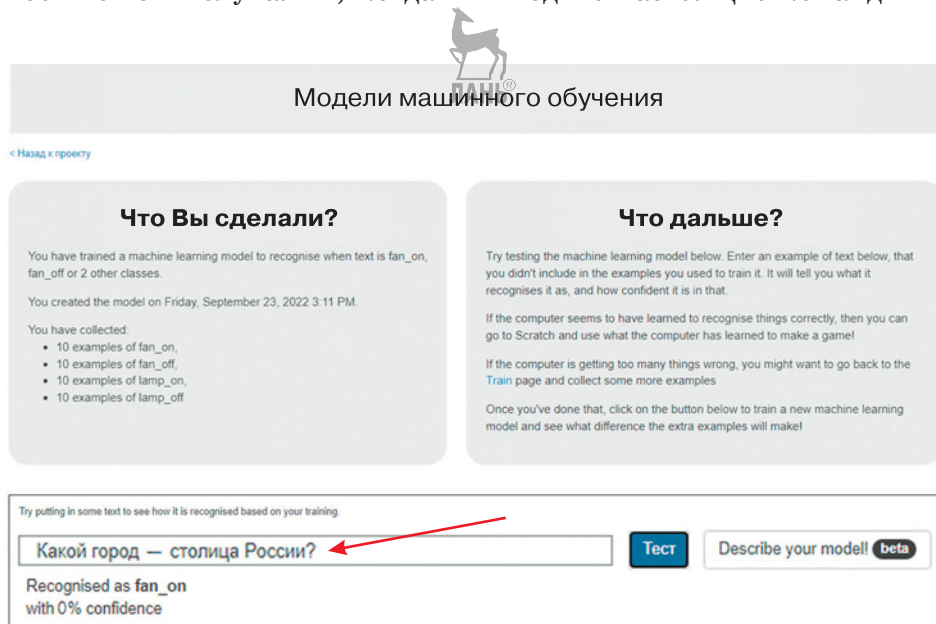


Рис. 10.13. Тестирование вашего умного помощника

включению или выключению вентилятора или настольной лампы. А какую степень уверенности показывает ваша модель, когда дает верный ответ?

Как только вы поймете, что ваша модель машинного обучения показывает правдоподобную оценку степени уверенности в самых разных случаях, добавьте этот показатель в свой Scratch-проект. Обновите скрипт так, чтобы он соответствовал рисунку 10.14.

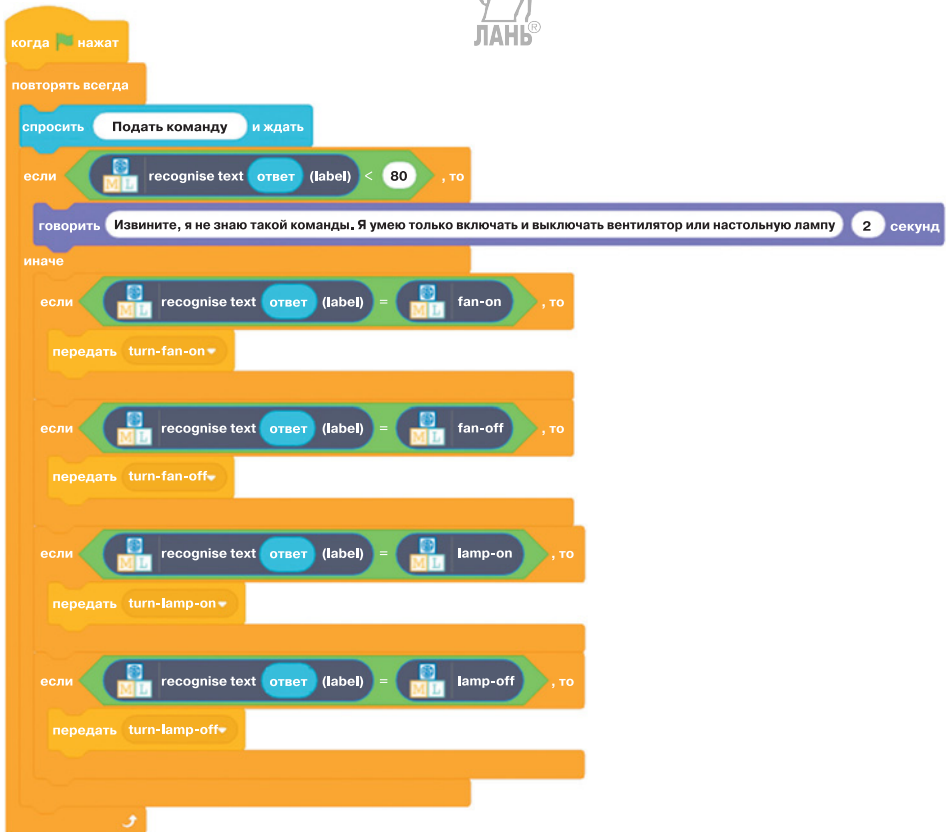


Рис. 10.14. Использование показателя степени уверенности в проекте

Теперь, если модель не уверена в своем ответе хотя бы на 80%, ни вентилятор, ни лампа не будут реагировать, а на экране в течение двух секунд будет отображаться сообщение «Извините, я не знаю такой команды. Я умею только включать и выключать вентилятор или настольную лампу».

Также вы можете изменить значение 80% на любое другое, которое больше подходит вашей модели машинного обучения по результатам тестирования.

А что еще можно сделать, чтобы улучшить проект?

Говорить, вместо того чтобы набирать текст вручную

Вы можете сделать свой проект еще более похожим на настоящих умных помощников. Обычно они используют голосовые команды вместо текстовых (а иногда и вместе с ними).

Нажмите на кнопку «Добавить расширение» в нижней части Панели инструментов (кнопка выглядит как два блока и знак «+» между ними). В открывшейся библиотеке расширений выберите расширение «Текст в речь» и измените свой скрипт так, как показано на рисунке 10.15*.

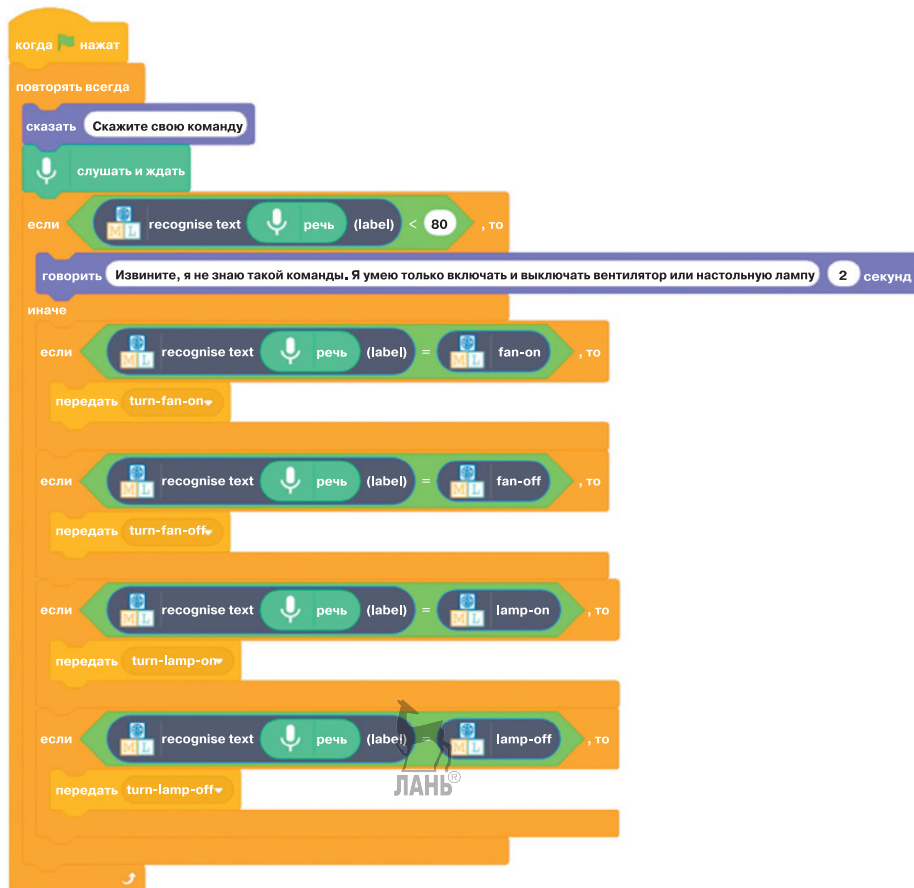


Рис. 10.15. Добавьте своему умному помощнику возможность распознавания речи

Отлично! Теперь ваш умный помощник совсем как настоящий! Но что же можно сделать, чтобы он стал еще лучше?

* На момент написания книги расширение «Текст в речь» работало корректно только в веб-браузере Google Chrome. — *Прим. ред.*

Сбор обучающих примеров



Машинное обучение часто используется для распознавания текста, потому что это быстрее и удобнее, чем программировать пошаговые инструкции. Но для того чтобы действительно хорошо обучить модель, требуется огромное количество обучающих примеров. Чтобы создавать профессиональные системы, гораздо более сложные, чем в наших проектах, нужен более эффективный способ сбора обучающих примеров, чем добавлять каждый из них вручную. А что, если вместо того, чтобы попросить одного человека добавить 100 обучающих примеров, попросить 100 человек добавить по одному примеру? Или тысячу человек? Или десятки тысяч человек?

Если вы поймете, где именно ваша модель ошибается, имеет смысл собрать больше обучающих примеров на эту тему. Но что, если модель машинного обучения показывает очень низкую степень уверенности? Или, например, пользователь продолжает давать одну и ту же команду, но каждый раз ее немножко по-другому формулирует? Это может привести к тому, что модель машинного обучения будет неправильно распознавать команды или не делать то, чего от нее требует человек. И это полезная *обратная связь* для вас как для разработчиков. Что, если пользователь нажмет кнопку «Мне не нравится это приложение»? Что, если он в конечном итоге перестанет что-либо нажимать? Что, если он будет говорить все более и более раздраженно? Это все обратная связь, которая вам на самом деле необходима.

Есть много способов понять, что что-то пошло не так. И каждый раз, когда это происходит, вы получаете пример, который можно добавить в одну из коллекций обучающих примеров и улучшить свою модель машинного обучения.

В реальных проектах используются сразу все возможные способы сбора обучающих примеров — и получение примеров от большого количества людей, и получение обратной связи от пользователей, и многое другое. Все это способствует тому, чтобы компьютеры лучше понимали нас.

Что вы узнали



В этой главе вы узнали, как можно использовать машинное обучение для распознавания команд в тексте и как это можно использовать для создания систем, которые будут понимать, чего мы от них хотим, и следовать нашим инструкциям.

В этом проекте вы использовали машинное обучение точно так же как это делают умные помощники, такие как Алиса от компании Яндекс, Google Home, Alexa от компании Amazon, Cortana от компании Microsoft и Siri от компании Apple. Интерфейс естественного языка позволяет нам говорить нашим устройствам, что мы хотим от них, вместо того чтобы набирать текст на клавиатуре или нажимать на кнопки.

Когда вы говорите своему смартфону установить будильник, сообщить точное время или воспроизвести вашу любимую песню, он должен классифицировать ваше намерение, т. е. распознать конкретную команду. То есть он должен распознать каждое слово, а затем определить среди них намерение.

Производители смартфонов и разработчики умных помощников обучили модель машинного обучения распознавать значение пользовательских команд. Они создали список категорий — все возможные команды, которые, по их мнению, могут давать пользователи. И затем для каждой из категорий они собрали множество обучающих примеров того, как разные пользователи могут формулировать такие команды.

И в вашем проекте, и в больших реальных проектах процесс создания умных помощников в целом выглядит одинаково. Он состоит из следующих этапов:

1. Определить (предугадать) набор команд, которые могут исходить от пользователей.
2. Собрать обучающие примеры для каждой из команд.
3. Использовать эти примеры для обучения модели машинного обучения.
4. Создать скрипт или написать программу для тех действий, которые компьютер должен выполнять по результатам распознавания каждой из команд.

Однако разница состоит в том — захотите ли вы создать настоящего умного помощника. Если да, то вам нужно будет повторить все эти шаги для тысяч команд, а не для четырех, как в этом проекте.

В следующей главе вы будете использовать эту возможность для создания программ, способных отвечать на вопросы пользователя, — чатботов.



ГЛАВА 11

Чатботы

В предыдущей главе вы узнали о классификации намерений — о создании систем машинного обучения, которые могут понимать значение текста, т. е. распознавать в тексте команды (намерения). Вы узнали, что чаще всего классификация намерений используется для создания различных умных помощников, которые могут распознавать наши команды (голосовые или текстовые) и выполнять соответствующие действия. Вы даже создали собственного «умного помощника»!

В этой главе вы узнаете, где еще используются модели машинного обучения, которые способны понимать значение текста и распознавать команды. Речь идет о **вопросно-ответных системах** (QA-системах, от англ. Question-Answering system). Вопросно-ответные системы могут распознавать вопросы пользователей и давать на них ответы, которые они автоматически подбирают из встроенного банка данных.

В отличие от поисковых систем, которые возвращают список веб-страниц, возможно, содержащих нужный ответ, вопросно-ответные системы возвращают конкретный ответ на конкретный вопрос. Это сложнее, так как вопросно-ответные системы должны гораздо лучше и глубже «понимать» как сам вопрос, так и содержание веб-страниц или документов, которые могут содержать ответы. Например, если пользователь задаст вопрос: «Как звали Пушкина?», то ответ должен быть: «Александр Сергеевич», а не список статей, содержащих биографии великих поэтов.

Вопросно-ответные системы — результат многолетних исследований в области искусственного интеллекта. С 1999 года Национальный институт стандартов и технологий США (NIST) ежегодно проводит конкурс, на котором образовательные учреждения и коммерческие организации представляют свои вопросно-ответные системы и соревнуются, чья разработка правильно ответит на наибольшее количество вопросов.

Пожалуй, одна из наиболее известных вопросно-ответных систем — Watson от компании IBM. В главе 1 вы уже узнали, что Watson принял участие в американской телевизионной викторине «Jeopardy!» и победил двух чемпионов. И это притом что викторина «Jeopardy!» известна своими сложными вопросами (часто даже с подвохом) из огромного количества самых разных областей, а значит, для компьютера она изначально сложнее, чем для человека.

В этой главе вы узнаете о чатботах. **Чатботы** — это программы, которые имитируют человеческое общение. По сравнению с вопросно-ответными системами, способными играть в викторину «Jeopardy!», чатботы являются даже более простой задачей для компьютера.

Во-первых, чатботы обычно создаются для ответов на вопросы на довольно ограниченный набор конкретных тем. В то же время вопросно-ответные системы должны уметь отвечать на самые разнообразные вопросы на любую тему (и эту тему угадывать!).

Во-вторых, чатботы часто используют заранее подготовленные ответы. От простых чатботов никто не ждет, что они будут искать ответы самостоятельно. Сложные чатботы могут иметь более сложные сценарии поведения и задавать наводящие или уточняющие вопросы, но в итоге принцип действия у них тот же.

С каждым днем чатботы становятся все более и более популярными. Их часто используют компании для общения с клиентами, особенно когда нужно отвечать на типовые вопросы о продуктах или услугах компании. В этом случае на наиболее часто задаваемые вопросы может ответить чатбот, а более сложные вопросы можно перенаправить сотруднику службы поддержки.

Вы наверняка уже видели чатботы на многих сайтах и в мобильных приложениях. Они могут делать что угодно — помогать вам оформить заказ пиццы, подобрать нужный размер одежды и аксессуаров, ответить на вопросы о погоде, назначить встречу, забронировать столик в ресторане, следить за состоянием вашего здоровья и многое другое.

В этом проекте вы создадите свой собственный чатбот и научите его отвечать на вопросы на любую тему, которую выберете сами. Пример моего проекта представлен на рисунке 11.1. Выполнять этот проект вы будете по уже знакомому из главы 10 алгоритму:

1. Определить (предугадать) набор вопросов, которые могут задавать пользователи.
2. Собрать обучающие примеры для каждого из вопросов.
3. Использовать эти примеры для обучения модели машинного обучения.

4. Подготовить ответы, которые компьютер должен выдавать по результатам распознавания каждого из вопросов.

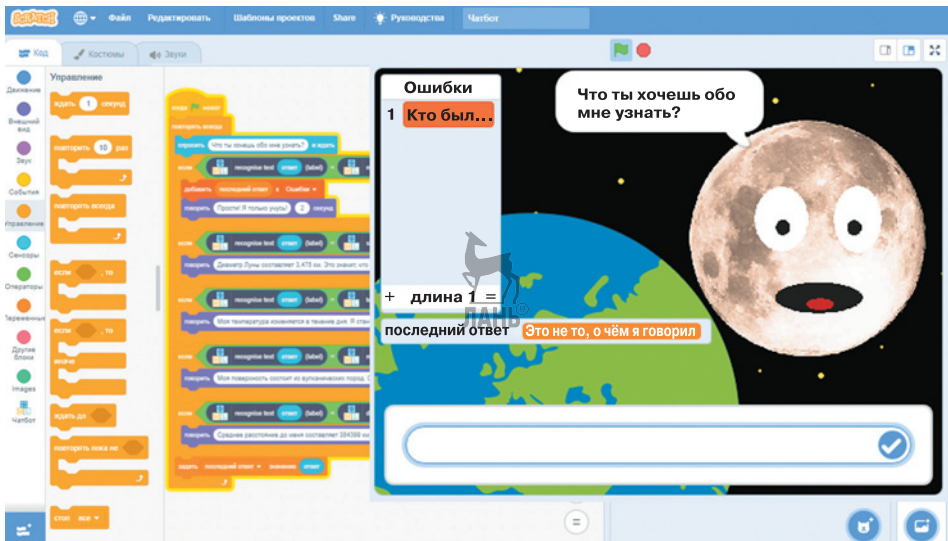


Рис. 11.1. Чатбот использует модель машинного обучения, чтобы отвечать на вопросы пользователя

Давайте же скорее начнем! Будет здорово!

Выполнение проекта

Для начала вам нужно выбрать тему, на которую ваш чатбот должен будет научиться общаться с пользователями. Это может быть абсолютно любая тема, но если вы не смогли с ходу выбрать только одну, вот вам мои рекомендации:

- ваша любимая книга;
- ваш любимый сериал;
- ваша любимая спортивная команда;
- ваш любимый актер, музыкант, писатель;
- космос, планеты, Солнечная система;
- динозавры;
- любая историческая эпоха, например Древняя Греция или Средневековье.

В моем примере я создал чатбот на тему космоса. Он отвечает на вопросы о Луне.

Подготовка персонажа

Зайдите в Scratch через сайт нашего проекта <https://machine-learning forkids.co.uk/scratch3/>. Здесь вам нужно будет добавить фон, а также нарисовать главного персонажа для вашего чатбота.

Убедитесь, что общая картинка, которую вы создаете, действительно соответствует выбранной теме. Например, если ваш чатбот должен отвечать на вопросы о Римской империи, вы можете изобразить центуриона на фоне поля битвы.

Мой чатбот должен отвечать на вопросы о Луне, поэтому я выбрал фон на тему космоса (рис. 11.2).

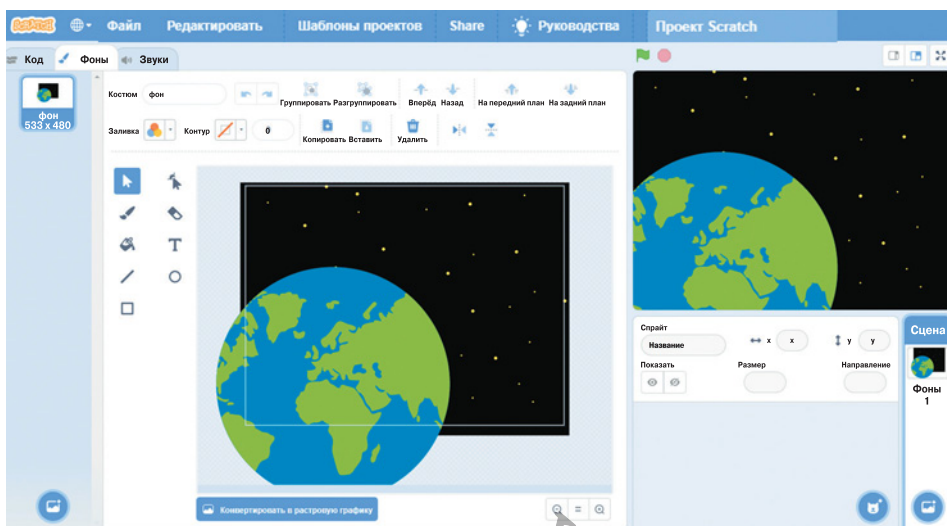


Рис. 11.2. Создание фона для чатбота

Для главного персонажа моего чатбота я создал спрайт Луны и дорисовал ему мультяшные глаза и рот (рис. 11.3).

Если вы не любите рисовать, можете выбрать уже готовые варианты. Наведите указатель мыши на иконку «Выбрать фон» или «Выбрать спрайт» в правом нижнем углу экрана. Там вы можете выбрать один из предустановленных шаблонов или загрузить изображение со своего компьютера, как вы уже делали в предыдущих проектах. Например, если вы создаете чатбот на тему своей любимой музыкальной группы, можете загрузить ее фото. В том случае, если вы создаете чатбот для спортивной команды или компании, можете загрузить ее логотип.

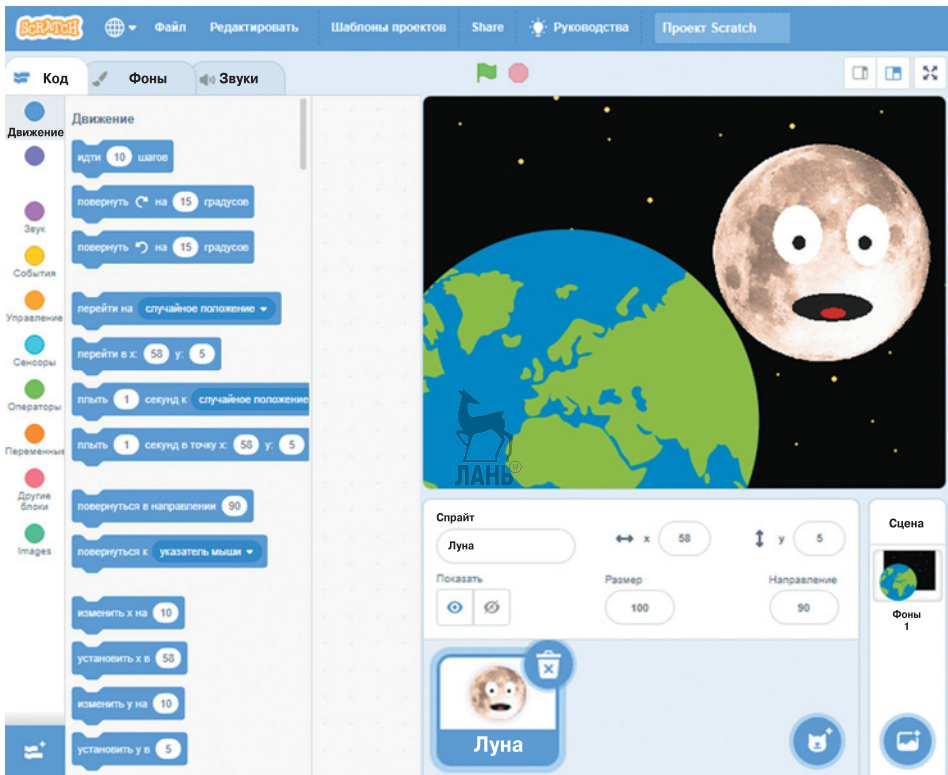


Рис. 11.3. Создайте спрайт для главного персонажа вашего чатбота



Примечание

Если вы не помните, как создать фон в Scratch, вернитесь к шагу 6 главы 3 (с. 47). Если вы не помните, как создать спрайт, вернитесь к шагу 9 главы 5 (с. 78).

Когда закончите выбирать фон и создавать главного персонажа, не забудьте сохранить свой проект Scratch. Тогда все изменения запомнятся и вы сможете продолжить работу в любой момент позже. Если вы не помните, как это сделать, во введении этой книги есть раздел «Сохранение результатов вашей работы» (с. 17).

Обучение модели

1. Создайте новый проект машинного обучения и назовите его «Чатбот». В качестве распознаваемого типа данных выберите «текст».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2. Желательно создавать проект без входа в профиль (в режиме гостя).

2. Нажмите на кнопку «Обучить», как показано на рисунке 11.4.

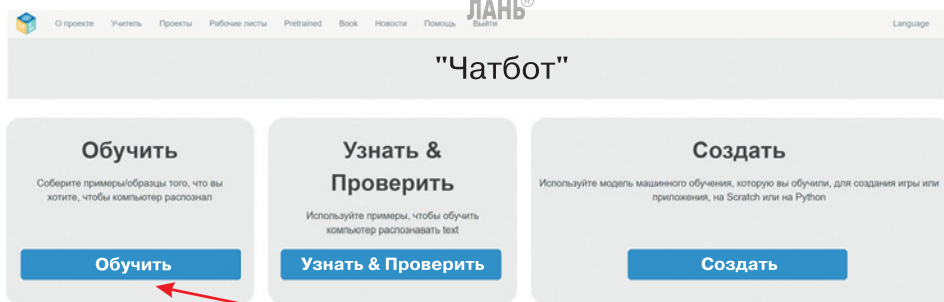


Рис. 11.4. Переход к первому этапу проекта машинного обучения — сбору обучающих примеров

3. Подумайте, какой самый популярный вопрос могут задавать люди на тему, которую вы выбрали.

Я выбрал тему вопросов о Луне, и мне кажется, самый популярный вопрос о Луне — это насколько она большая.

Когда вы определитесь с первым вопросом, нажмите на кнопку «Добавить новую метку» (рис. 11.5) и создайте первую коллекцию обучающих примеров, подобрав ей такое имя (в одно

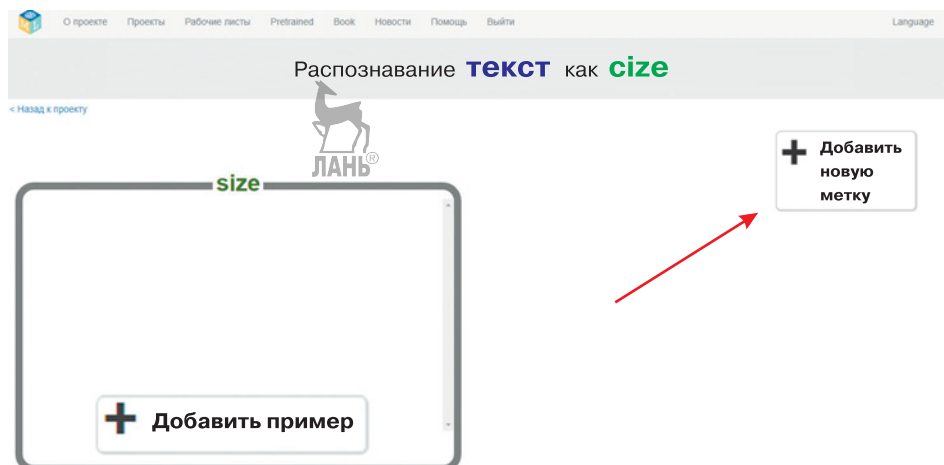


Рис. 11.5. Создание коллекции обучающих примеров — ответов на первый вопрос

слово), которое ассоциировалось бы с этим вопросом. Например, свою первую коллекцию я назвал «size» (англ.: размер).

4. Нажмите на кнопку «Добавить пример» и заполните первую коллекцию примерами того, как можно сформулировать выбранный вами вопрос (рис. 11.6).

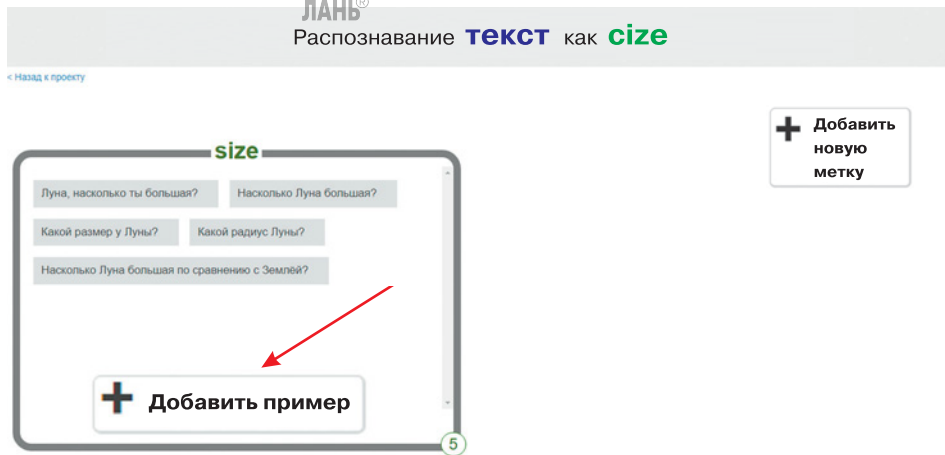


Рис. 11.6. Добавьте примеры того, как можно сформулировать первый вопрос

Подумайте, как разные люди могут формулировать один и тот же вопрос. Постарайтесь заполнить коллекцию самыми разнообразными фразами. При этом вам пока не надо думать об ответе на этот вопрос. Сейчас вас интересует только сам вопрос.

Добавьте столько вариантов этого вопроса, сколько сможете придумать. Только не забывайте, что в коллекции должно быть не менее пяти обучающих примеров. Эти примеры будут использоваться для обучения модели, которая сможет распознать, задал ли пользователь этот вопрос или какой-нибудь другой, и дать на него ответ.

Если главный персонаж вашего проекта Scratch олицетворяет выбранную тему, добавьте в коллекцию несколько примеров, в которых вопрос адресован персонажу. Например, мой главный персонаж — Луна с мультяшным лицом, которая отвечает на вопросы о Луне. Поэтому я добавил такие примеры как «Луна, насколько ты большая?». 

5. Отлично! А теперь придумайте еще несколько других наиболее популярных вопросов на вашу тему. Затем снова нажмите на кнопку «Добавить новую метку» и создайте коллекцию для каждого из этих вопросов. После этого с помощью кнопки «Добавить пример» заполните коллекции обучающими примерами —

фразами, с помощью которых можно задавать эти вопросы. Как и всегда, вам нужно собрать не менее пяти примеров в каждую коллекцию. Не забывайте: фразы должны быть разнообразными и некоторые из них могут содержать обращение к главному персонажу.

В моем примере я выбрал четыре вопроса и собрал по пять обучающих примеров в коллекции каждого из вопросов (рис. 11.7). Ваш проект может сильно отличаться от моего в зависимости от того, какую тему и какие вопросы вы выбрали.

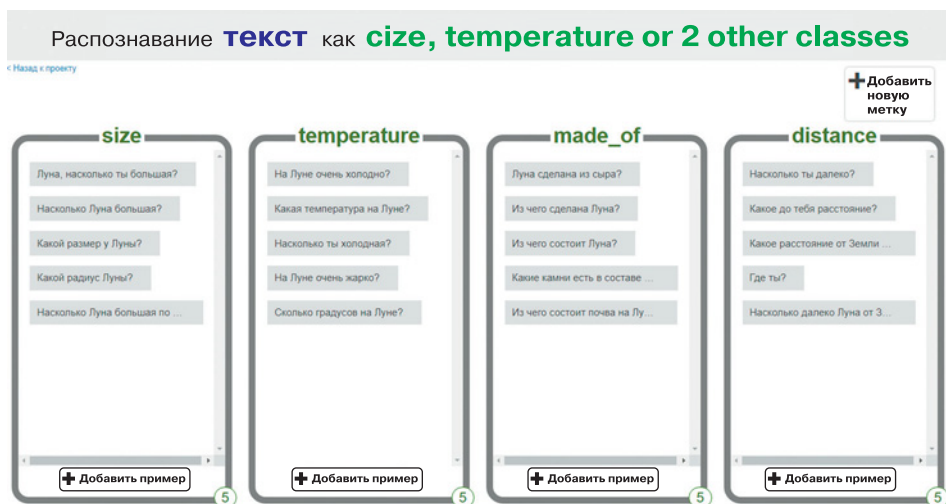


Рис. 11.7. Добавьте обучающие примеры для каждого из вопросов, стараясь делать их максимально разнообразными

В вашем проекте вы попытаетесь угадать, какие самые популярные вопросы могут задавать люди на выбранную вами тему и как именно они их могут задавать. В реальных же проектах машинного обучения такие обучающие примеры могут быть получены от настоящих клиентов или пользователей, поэтому компьютер получает действительно самые популярные вопросы и самые распространенные способы формулировок этих вопросов. Например, если магазин электроники хочет создать виртуального помощника, чтобы отвечать на вопросы о настройке телевизоров, он может использовать вопросы, которые клиенты задавали на эту тему по телефону или по электронной почте. Если банк хочет создать виртуального помощника, чтобы отвечать на вопросы о счетах и вкладах, он может использовать вопросы, которые задавали клиенты в чате на сайте этого банка. То есть такие компа-

нии могут собрать в качестве обучающих примеров для моделей машинного обучения реальные вопросы, которые задавали реальные люди.

В главе 3 вы убедились, что модель машинного обучения работает лучше, если обучающие примеры похожи на те, что будут использоваться в реальном проекте. Вы узнали, что если вы хотите, чтобы модель распознавала фотографии какого-то животного, то ее сначала нужно обучать на фотографиях этого животного. А если модель должна распознавать это животное, но в виде героя мультфильма, модель нужно обучать на примерах изображений из мультфильмов.

Модели машинного обучения, которые распознают текст, работают так же. Если обучить их на примерах вопросов, которые задают реальные люди, модели будут давать более точные ответы. Сейчас, пока учитесь, вы можете сами придумать обучающие примеры, но в реальных больших проектах лучше *собирать* для использования моделью машинного обучения формулировки примеров, а не *придумывать* их самим.

6. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

7. Нажмите на кнопку «Узнать и проверить», чтобы перейти к следующему этапу проекта (рис. 11.8).

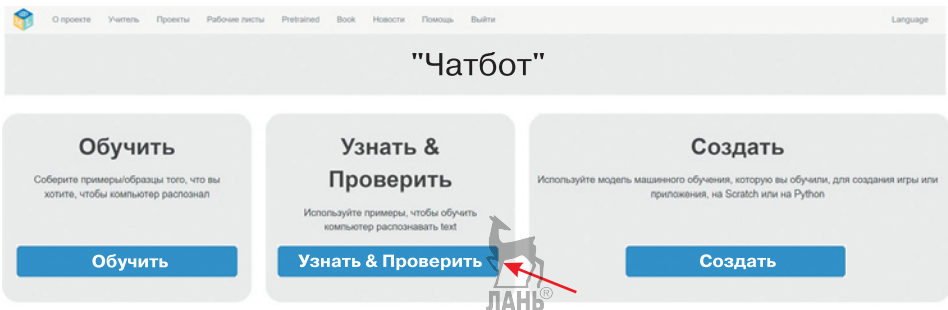


Рис. 11.8. Переход ко второму этапу работы над проектом — этапу «Узнать и проверить»

8. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 11.9).

Компьютер будет использовать собранные вами примеры для того, чтобы научиться распознавать наиболее популярные вопросы, которые люди могут задавать на выбранную вами тему. Это может занять несколько минут, наберитесь терпения. Компьютер

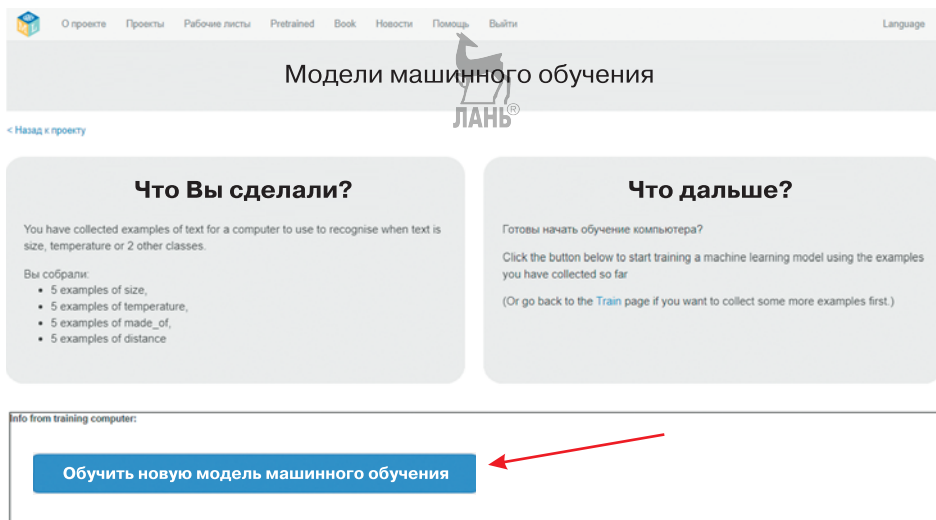


Рис. 11.9. Обучение модели машинного обучения с использованием примеров, которые вы собрали

использует это время на обдумывание, что может быть общего у примеров в каждой из коллекций. Например, какие слова используются, чтобы задать каждый из вопросов, как строятся фразы, насколько они длинные или короткие и так далее — любые существующие закономерности.

Подготовка проекта

Ура! Теперь у вас есть тема, главный персонаж и даже модель машинного обучения. Это все что нужно, чтобы создать ваш первый чатбот.

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

2. Нажмите на кнопку «Создать» (рис. 11.10).

3. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch. На Панели инструментов вы увидите новую категорию блоков, которая будет называться так же, как и ваш проект.

4. В Главном меню выберите **Файл** → **Загрузить с компьютера**, как показано на рисунке 11.11, и загрузите проект, который вы недавно сохраняли, со Сценой и главным персонажем вашего чатбота.

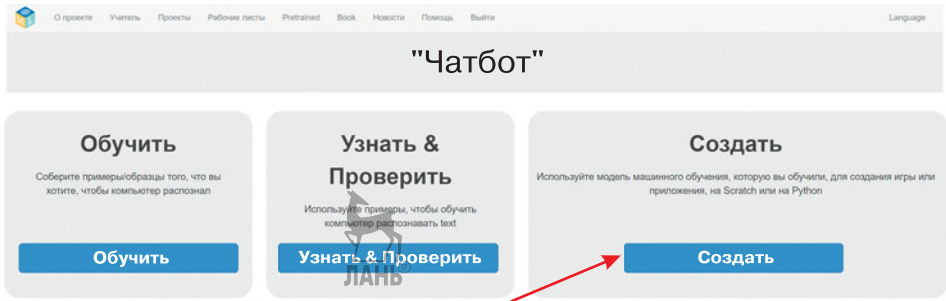


Рис. 11.10. Переход к третьему этапу проекта — этапу «Создать»

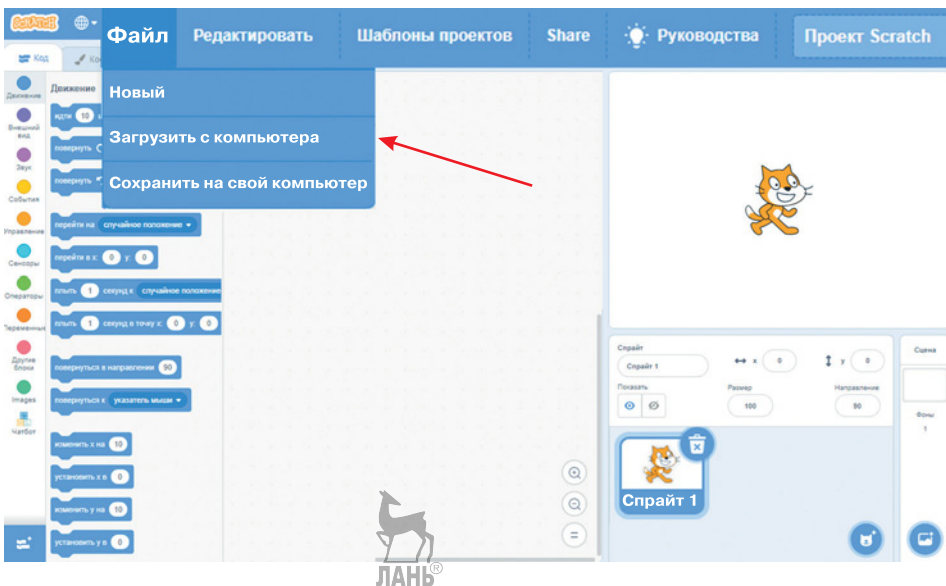


Рис. 11.11. Откройте проект со Сценой и главным персонажем, который вы сохранили на свой компьютер ранее

5. Теперь создайте в этом проекте такой же скрипт, как на рисунке 11.12.

Этот скрипт просит пользователя ввести (набрать на клавиатуре) вопрос, а затем использует вашу модель машинного обучения для того, чтобы распознать, что именно спросил пользователь.

Имейте в виду, что ваш скрипт, скорее всего, будет отличаться от моего, потому что я работал с вопросами о Луне и именно на этих вопросах обучал свою модель машинного обучения. А вам нужно встроить в скрипт вопросы именно на вашу тему, поэтому будьте внимательны.

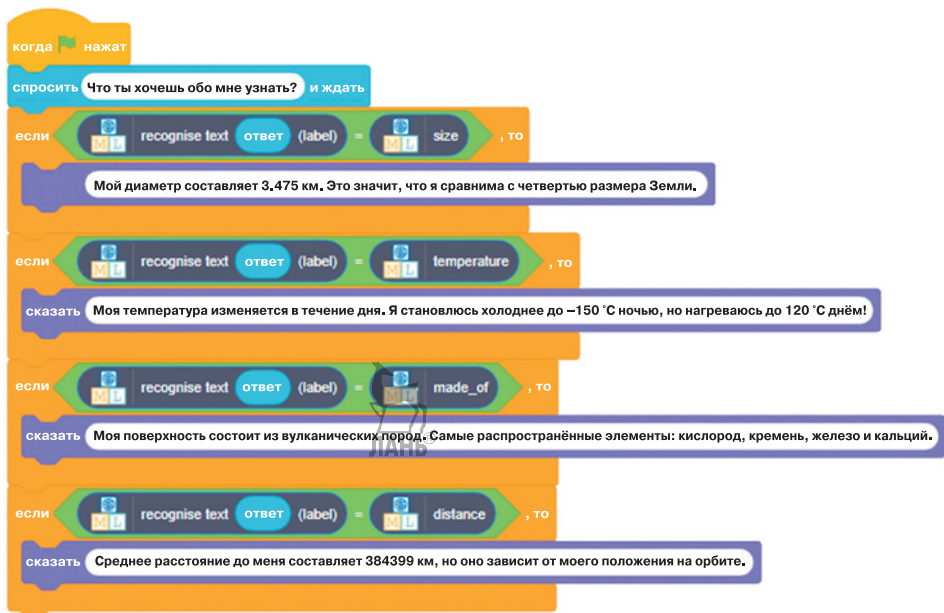


Рис. 11.12. Скрипт для создания простого чатбота

Вам также нужно будет добавить в скрипт ответы на эти вопросы. Если вы не знаете ответ на какой-то из вопросов — самое время провести небольшое исследование и получить все ответы. Теперь ваш чатбот будет знать, что ответить своим пользователям!

6. Следующим шагом нужно обязательно сохранить все внесенные изменения. Для этого в Главном меню выберите **Файл** → **Сохранить на свой компьютер**.

Тестирование модели

Нажмите на зеленый флажок, чтобы запустить проект и проверить вашу модель машинного обучения в деле. Попробуйте ввести несколько вопросов на выбранную вами тему. Что же ваша модель машинного обучения? Дает правильные ответы?

Обзор и улучшение проекта

Поздравляю! Вы только что создали свой первый чатбот. Он может распознавать популярные вопросы на выбранную вами тему и отвечать на них. Здорово, не правда ли?

Но что же можно сделать, чтобы его улучшить? Давайте посмотрим!

Реакция на пользовательские сообщения об ошибках

Системы искусственного интеллекта просто не могут распознавать все и всегда правильно. Поэтому вы можете научить свой проект «учиться на ошибках». Самый простой способ это сделать — научить вашу модель машинного обучения отслеживать моменты, когда пользователи недовольны результатами ее работы.

Вернитесь к этапу обучения и добавьте в свой проект машинного обучения новую коллекцию обучающих примеров. Назовите ее «mistake» (англ.: ошибка). Наполните эту коллекцию примерами фраз, которые могут произносить пользователи, когда не удовлетворены ответом, который дала модель (рис. 11.13).

Например, в эту коллекцию можно добавить такие фразы, как «Нет, это не то, что я имел в виду».

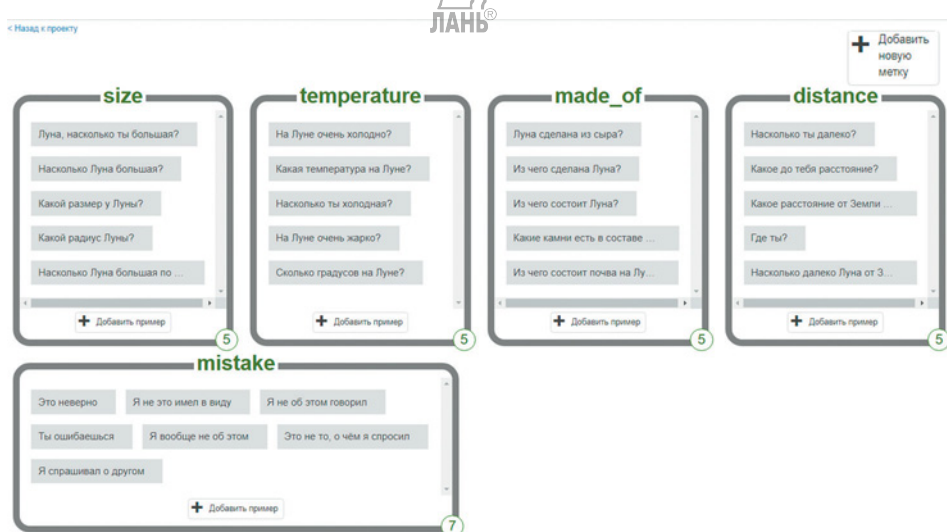


Рис. 11.13. Сбор обучающих примеров — фраз, которые могут писать пользователи, когда они недовольны работой модели

После того как вы добавите в эту коллекцию хотя бы пять обучающих примеров, не забудьте вернуться к этапу «Узнать и проверить», чтобы обучить модель машинного обучения заново.



Примечание

Вам также нужно будет закрыть и снова открыть ваш проект в Scratch, чтобы он смог использовать примеры из новой коллекции. Но перед тем как закрывать, убедитесь, что вы сохранили все внесенные изменения. Это очень важно, ведь иначе вам придется создавать скрипт заново.

Теперь ваш чатбот сможет отследить момент, когда пользователь недоволен его работой. Но как ему на это реагировать? Самый простой способ — попросить прощения. Для этого обновите скрипт так, как показано на рисунке 11.14.

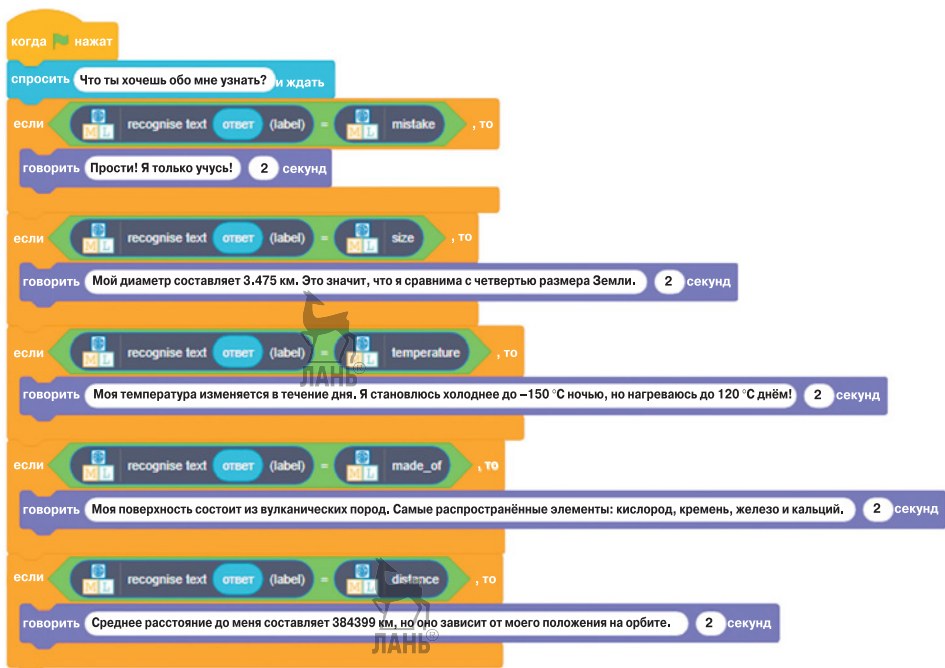


Рис. 11.14. Обновите свой скрипт так, чтобы чатбот мог приносить извинения пользователю в случае ошибки

Кстати, вы можете сделать свой проект еще лучше, если будете вести запись ошибок. Например, я создал для этого новый список (как вы уже делали в главе 8) с именем «Ошибки». Чтобы это сделать, перейдите на вкладку «Код», выберите категорию «Переменные» на Панели инструментов и нажмите на кнопку «Создать список». Затем нужно обновить скрипт так, как показано на рисунке 11.15. Теперь, если пользователь будет недоволен полученным ответом, его вопрос будет добавлен в список ошибок.

В моем примере, когда я задал чатботу вопрос: «Кто был первым человеком на Луне?», он выдал ответ, указывающий расстояние, на котором Луна находится от Земли. На это я ответил: «Нет, это не то, о чем я спросил», и мой первый вопрос о человеке на Луне был добавлен в список ошибок.

Запись ошибок — это довольно популярный способ улучшения настоящих больших проектов машинного обучения. Список оши-

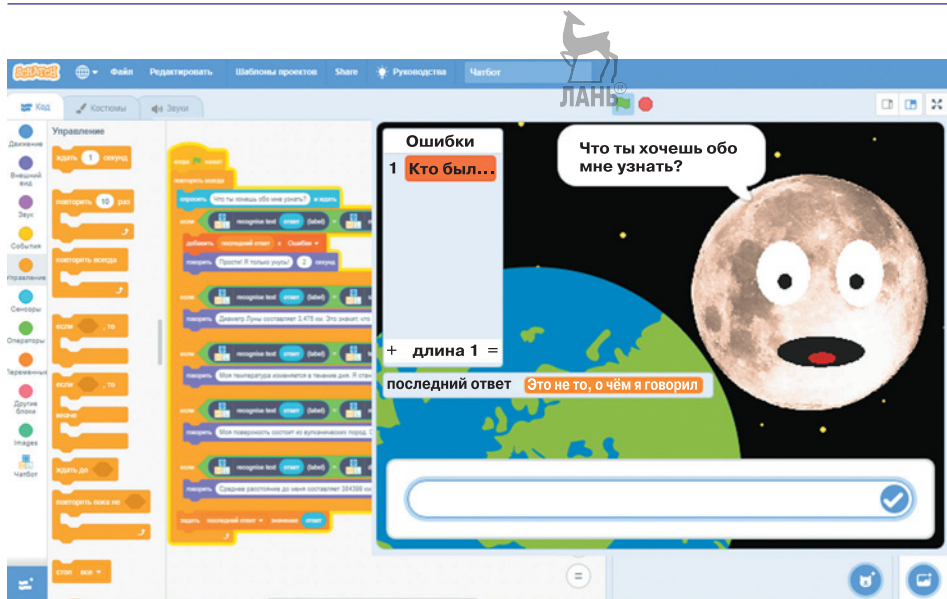


Рис. 11.15. Ведение записи ошибок (вопросов, на которые чатбот отвечает неверно)

бок обычно добавляется в новую коллекцию обучающих примеров и используется для обучения модели при выпуске следующей версии проекта.

Но что еще можно сделать, чтобы улучшить этот проект?

Распознавание пользовательского недовольства

На самом деле пользователи вашего чатбота не всегда смогут или захотят сообщить вам, если ваша модель машинного обучения делает что-то не так. Тогда какие еще есть способы узнать, что пользователи чем-то недовольны? Ведь для вас как для разработчиков это очень важно!

В главе 7 вы научили свою модель машинного обучения распознаванию эмоций в тексте и распознаванию тональности текста. Для данного проекта это может быть очень полезным, поскольку тогда чатбот может научиться распознавать случаи, когда пользователь злится или становится недовольным.

Для этого вам понадобятся сразу две модели машинного обучения в одном проекте. Одна модель должна быть обучена распознавать содержание текста вопроса (это вы только что сделали для своего чатбота). Другая же модель должна быть обучена распознавать эмоции в тексте вопроса (так, как вы делали в про-

екте главы 7). И если вторая модель будет показывать высокую степень уверенности, что в вопросе пользователя присутствует раздраженность, чатбот должен принести извинения, вместо того чтобы снова пытаться ответить на этот вопрос.

Распознавание эмоций пользователя в тексте и отображение сообщения с извинениями, когда это необходимо, — это часто используемый прием в реальных проектах. Особенно если это проекты из сферы услуг. Пользователи обычно недовольны, если современные технологии не понимают их и не выполняют того, что им нужно. Если система машинного обучения способна распознать, что в какой-то момент в общении с пользователем что-то пошло не так, и принести извинения, пользователь может перестать злиться и продолжить конструктивный диалог. Хорошо спроектированные системы предложат соединить пользователя с менеджером, который лично окажет содействие в решении проблемы. И это одно из лучших возможных решений.

А как насчет того, чтобы сделать ваш проект еще лучше?

Отвечать на вопросы только тогда, когда модель уверена в своем ответе

Распознавание случаев, когда пользователь начинает проявлять раздражение, — это отличная идея для проектов машинного обучения. Но гораздо лучшим решением было бы вовсе не допускать пользовательского недовольства! Ведь все программы написаны для того, чтобы облегчить жизнь пользователям, а не наоборот.

Вы можете использовать оценку степени уверенности распознавания вопроса, чтобы уберечь ваш чатбот от отображения неверных ответов.

В проекте главы 7 вы узнали, что степень уверенности — это показатель, который выражает в процентах, насколько модель машинного обучения уверена в своем ответе, независимо от того, на каких примерах она обучалась.

В этом проекте, если модель показывает низкую степень уверенности, это означает, что она не уверена, что поняла вопрос. Если вы измените скрипт так, как показано на рисунке 11.16, то сможете улучшить свой чатбот еще одним способом. Теперь, если модель не будет уверена, что распознала вопрос, чатбот принесет пользователю свои извинения, вместо того чтобы снова и снова давать неправильные ответы. А сам вопрос будет добавлен в список ошибок, чтобы использоваться в качестве обучающего примера при улучшении модели.

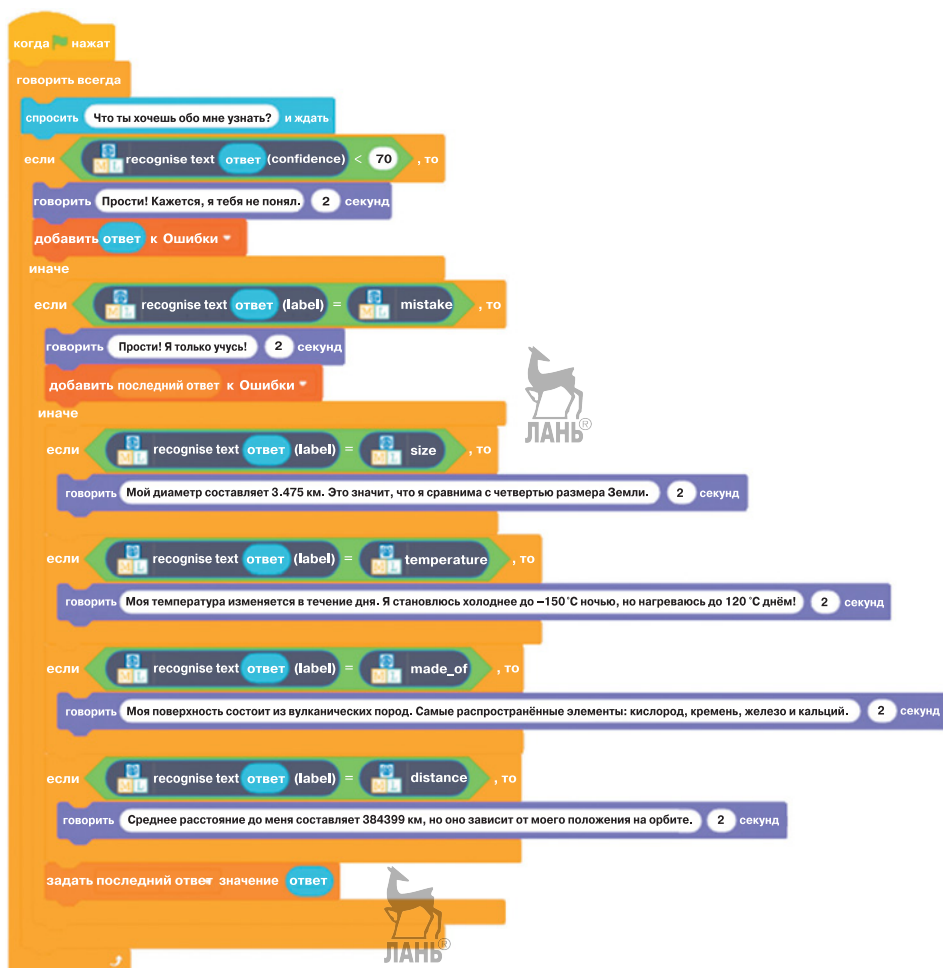


Рис. 11.16. Отслеживание вопросов с низкой степенью уверенности, добавление их в список ошибок и использование при следующем обучении модели

Этические вопросы использования машинного обучения

Все советы по улучшению вашего проекта взяты из настоящих больших проектов, использующих машинное обучение. Как вы уже убедились, сбор данных для обучающих примеров требует много времени и усилий. Чтобы сэкономить время и силы, многие компании собирают ровно столько обучающих примеров,

сколько достаточно для обучения модели и для того, чтобы проект просто заработал. После этого они запускают такой «сырой» проект.

Иногда такую версию проекта называют бета-версией, обращая внимание пользователя на возможное наличие в ней множества ошибок. Затем они собирают сообщения об ошибках и плохие отзывы пользователей и используют их в качестве обучающих примеров, чтобы улучшить модель машинного обучения и весь проект в целом. Например, это могут быть вопросы пользователей, на которые модель машинного обучения показала низкую степень уверенности, или же те вопросы, ответы на которые пользователи посчитали неверными или бесполезными. Тогда компании просматривают эти вопросы и распределяют их по соответствующим коллекциям обучающих примеров.

При этом компании утверждают, что делают это для того, чтобы их модели машинного обучения работали все лучше и лучше в будущем. Чем более разнообразные обучающие примеры они собирают и используют, тем более точные и подходящие ответы может давать модель.

Но иногда это удивляет и даже пугает пользователей, которые раньше не представляли, что все вопросы, которые они задают своим умным устройствам, могут быть записаны и переданы производителям этих устройств. Попробуйте поискать в Интернете новостные статьи на тему того, как производитель вашего любимого умного устройства прослушивает то, что пользователи говорят этому устройству в качестве команд (и не только). Как много статей на эту тему вам удалось найти? Что вы думаете о них и о реакции людей на них?

А как вы думаете, этично ли со стороны разработчиков собирать обучающие примеры из реальных разговоров пользователей или клиентов компании? Как вы считаете, должны ли они предупреждать своих пользователей о том, что будет вестись запись разговора? Должна ли быть предусмотрена ответственность разработчика за это? И как бы вы объяснили эту ситуацию пользователям, которые не представляют, что такое обучающие примеры и почему они так важны для машинного обучения? Как бы вы их успокоили? Подумайте об этом!*

* При первом запуске программы или устройства пользователь подписывает соглашение, в котором подробно описано, какие данные передает система и как они будут обрабатываться. Вся подобная информация передается в обезличенном виде, т. е. идентифицировать участников разговоров или владельцев странных привычек невозможно. — *Прим. ред.*



Что вы узнали

В этой главе вы узнали, что модель машинного обучения можно научить распознавать вопросы пользователей. Вы узнали разницу между вопросно-ответными системами и их более простыми аналогами — чатботами. Затем вы разработали свой собственный чатбот и обучили свою модель распознавать наиболее популярные вопросы на выбранную вами тему. После этого вы познакомились сразу с несколькими способами улучшения этого проекта и приближения его к реальным большим проектам. Вы узнали, как улучшить результативность и точность своего чатбота, например за счет отслеживания ошибок, ответа на вопрос в зависимости от настроения и эмоций пользователя, а также использования показателя степени уверенности для принятия решения о перенаправлении вопроса человеку (например, сотруднику службы поддержки). Наконец, вы узнали о некоторых этических вопросах, которые следует учитывать при обучении моделей машинного обучения на основе отзывов реальных людей.

В следующей главе вы попробуете сделать что-то новое и впервые обучите модель машинного обучения распознавать числа в упрощенной версии легендарной видеоигры «Pac-Man».





ГЛАВА 12

Создание игры «Убег от монстра»



Искусственный интеллект и машинное обучение имеют невероятный потенциал в еще одной сфере — разработке компьютерных игр. Особенно тех игр, в которых персонажи должны понимать ваши слова и действия. На сегодняшний день уже есть игры, в которых персонажи действуют по тому же принципу, что и чатбот, который вы создали в предыдущем проекте. Но тем не менее все еще остается простор для создания действительно «интеллектуальных» игр, которые адаптируются под действия игроков.

В этой главе вы рассмотрите обратный случай — не то, как искусственный интеллект может повлиять на развитие сферы разработки компьютерных игр, а как компьютерные игры могут повлиять на развитие искусственного интеллекта.

На самом деле игры — это отличная площадка для развития искусственного интеллекта. Ведь, если разобраться, они могут предоставить нам среду моделирования с четко определенными целью и задачами, способ сбора обучающих примеров и способ измерения эффективности системы машинного обучения. Все эти вещи делают игры отличной платформой для исследований в области разработки компьютерных систем, которые могут обучаться.

Есть одна классическая компьютерная игра, которая чаще всего используется в исследованиях в области искусственного интеллекта. Эта игра — «Рас-Ман» (ее аналог — «Ms. Рас-Ман»). Создан даже конкурс проектов искусственного интеллекта по игре «Ms. Рас-Ман», где ученые и исследователи представляют свои системы машинного обучения, которые соревнуются и выясняют — какая из них лучше играет в эту игру. Данный конкурс проводится уже несколько лет, начиная с 2007 года, и до сих пор используется в качестве задания для студентов, которые начинают изучать искусственный интеллект.

В этой главе вы обучите свою модель машинного обучения играть в упрощенную версию игры «Рас-Ман». Суть этой игры —

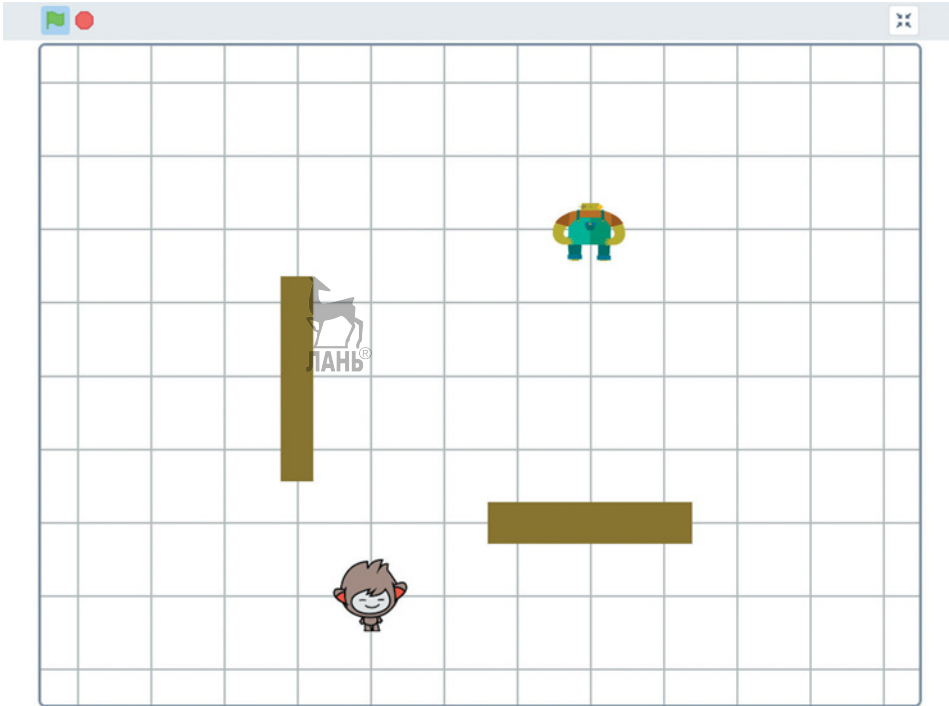


Рис. 12.1. Простая видеоигра, в которую научится играть ваша модель машинного обучения

управление персонажем внутри лабиринта, чтобы помочь ему убежать от монстра (рис. 12.1).

Звучит захватывающе, не так ли? Давайте же начнем!

Выполнение проекта

Прежде чем начать, попробуйте сами сыграть в эту игру, чтобы понять, что именно должен будет научиться делать компьютер. Для этого перейдите на страницу проекта <https://machinelearningforkids.co.uk/scratch3/> и выберите в Главном меню пункт **Шаблоны проектов** (рис. 12.2).

В открывшемся списке доступных шаблонов выберите проект с именем «Avoid the monster» (англ.: убеги от монстра), как показано на рисунке 12.3.

В этой игре вы будете играть за персонажа по имени «напо», который начинает ходить из левого нижнего угла Сцены. Ваша цель в этой игре — не быть пойманным монстром, который на-

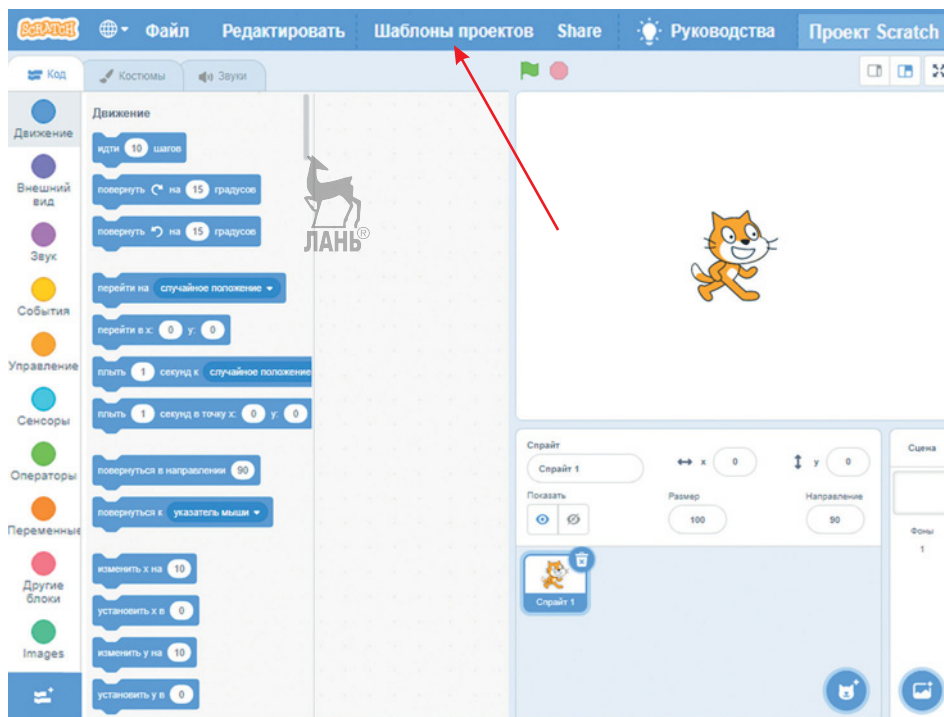


Рис. 12.2. Выберите пункт **Шаблоны проектов** для получения доступа к игре



Рис. 12.3. Выберите проект с именем «Avoid the monster» из библиотеки шаблонов Scratch

чинает ходить из правого верхнего угла Сцены. Убегать от него нужно так долго, как вы сможете.

Для управления персонажем папо используйте клавиши со стрелками на клавиатуре. Он может двигаться только по линиям сетки и только влево, вправо, вверх или вниз. Двигаться по диагонали папо не может.

Если вы не нажимаете ни на одну из стрелок, папо продолжает движение в том же направлении, в котором двигался после нажатия предыдущей клавиши.

Еще один важный момент, папо не может двигаться быстрее, чем монстр. В код игры добавлен таймер, поэтому и монстр, и папо могут совершать только один шаг в секунду.

Также на Сцене есть две стены. Ни монстр, ни папо не могут проходить сквозь стены, а только обходить их.

Теперь вы знаете правила. Нажмите на зеленый флажок, чтобы запустить игру, и попробуйте в нее поиграть.

Как долго вы смогли убежать от монстра?

Описание возможных состояний игры

Игровое поле можно представить не только как сетку (как вы это уже делали), но и как координатную плоскость или график (рис. 12.4).

Вы будете использовать эту координатную плоскость, чтобы описывать местоположение папо или монстра в виде пары коор-

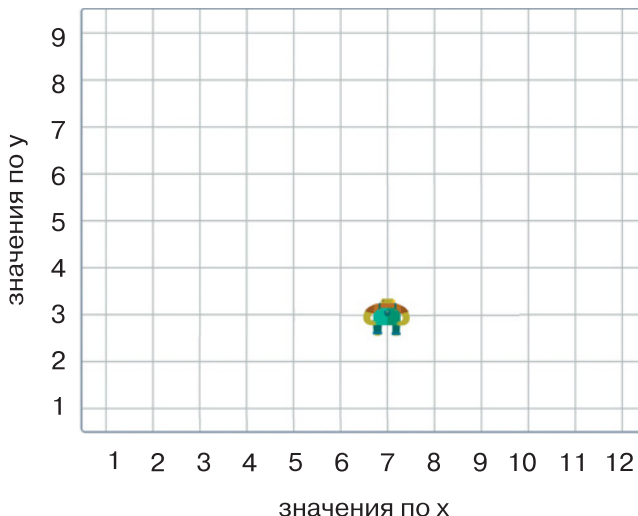


Рис. 12.4. Представление игрового поля в виде координатной плоскости с осями x и y

динат. Например, на рисунке 12.4 монстр находится в точке с координатами $x = 7$ и $y = 3$. Такая система обозначений позволит вам запрограммировать эту игру понятным для компьютера способом.

То есть, если вы дадите компьютеру четыре числа (координаты папо и координаты монстра), он сможет решить, куда должен двигаться папо, чтобы убежать от монстра.

Если компьютер получит координаты, которые показаны на рисунке 12.5, он должен решить, что папо нужно двигаться наверх, чтобы убежать от монстра.

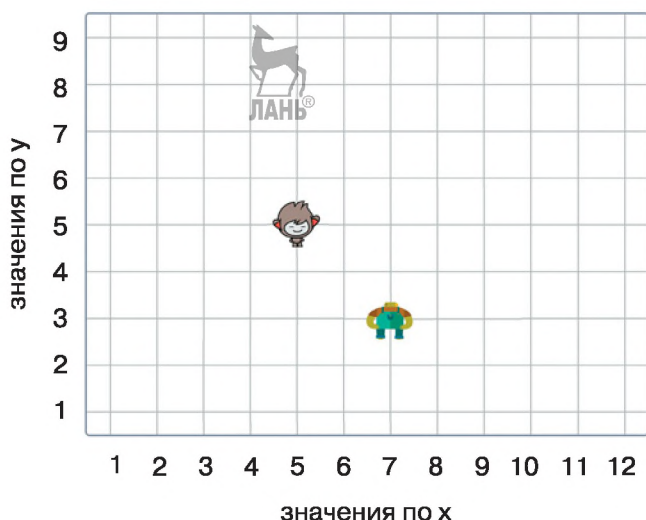


Рис. 12.5. Персонаж папо находится в точке с координатами $x = 5$ и $y = 5$, монстр находится в точке с координатами $x = 7$ и $y = 3$

Ваша задача в этом проекте — обучить модель машинного обучения определять лучшее направление движения, чтобы папо смог убежать от монстра.

Обучение модели

Чтобы научить компьютер играть в игру, вам нужно собрать примеры того, как в нее играть. Самый лучший способ — это играть в игру самому, а потом использовать свой игровой опыт для обучения компьютера.

Первый шаг — подготовка коллекции обучающих примеров, в которые вы будете добавлять примеры своих ходов.

1. Перейдите на страницу проекта <https://machinelearningforkids.co.uk/> Создайте новый проект машинного обучения, на-

зовите его «Убегите от монстра». В качестве распознаваемого типа данных выберите «числа».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Add a value» (англ.: добавить значение), в качестве имени значения укажите «nano x» (англ.: координата x персонажа nano), а в выпадающем списке «Тип значения» выберите «число». Затем нажмите на кнопку «Add another value» (англ.: добавить другое значение) и добавьте три новых значения с именами «nano y» (англ.: координата y персонажа nano), «monster x» (англ.: координата x монстра) и «monster y» (англ.: координата y монстра), как показано на рисунке 12.6. После этого нажмите на кнопку «Создать», чтобы создать проект и перейти к нему.

В этих значениях будут храниться координаты обоих персонажей (nano и монстра) после сделанных ходов.

Рис. 12.6. Подготовка значений для проекта «Игра „Убегите от монстра“»

3. Нажмите на кнопку «Обучить», как показано на рисунке 12.7.

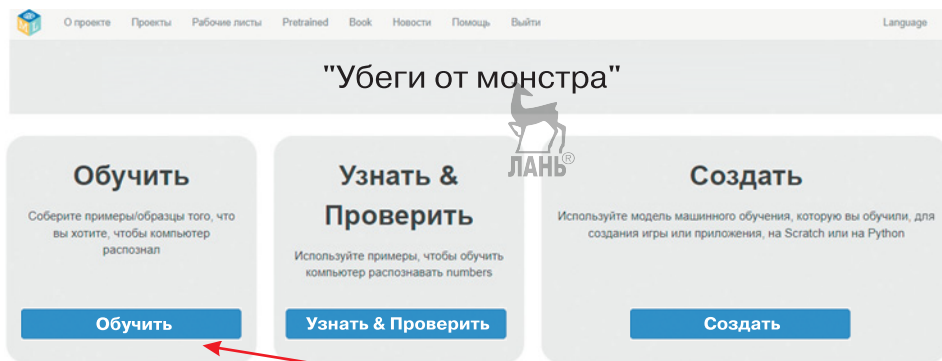


Рис. 12.7. Нажмите на кнопку «Обучить», чтобы начать подготовку коллекций обучающих примеров

4. Нажмите на кнопку «Добавить новую метку» (рис. 12.8) и создайте 4 коллекции — для 4 направлений, в которых может двигаться персонаж папо. Назовите их «go left» (англ.: ход влево), «go right» (англ.: ход вправо), «go up» (англ.: ход вверх) и «go down» (англ.: ход вниз). Символы «нижнее подчеркивание» между словами в названии коллекции будут добавлены автоматически.

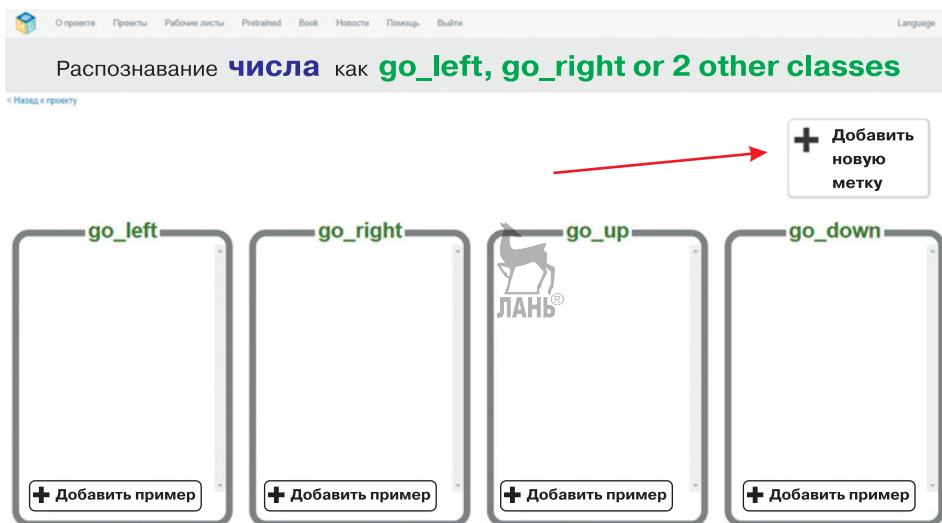


Рис. 12.8. Подготовьте четыре коллекции для сбора обучающих примеров с возможными ходами персонажа папо

Например, представьте, что вы играете в эту игру, и в данный момент папо находится в точке с координатами $x = 2$ и $y = 3$, а монстр находится в точке с координатами $x = 6$ и $y = 7$. Если вы нажмете на стрелку вправо, чтобы папо сделал ход вправо, то этот набор значений будет добавлен в коллекцию «go_right», как показано на рисунке 12.9.

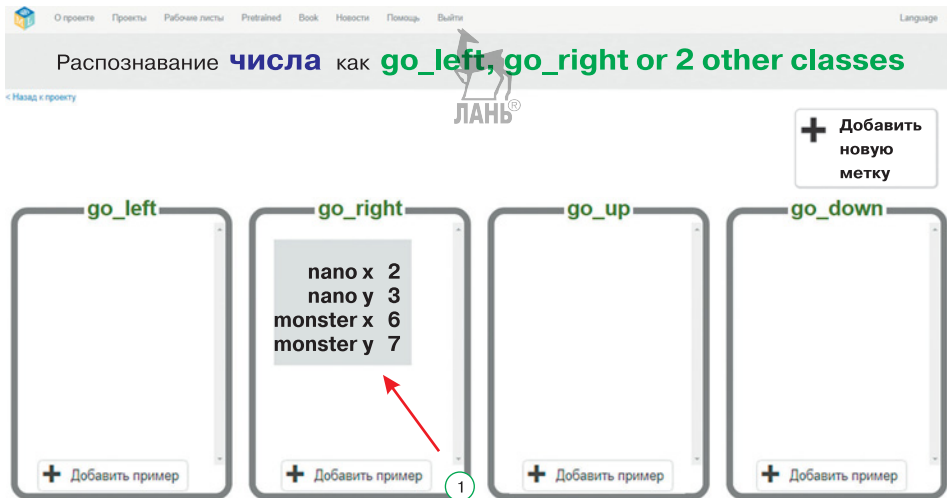


Рис. 12.9. Ходы, которые вы будете делать в игре, будут добавляться в коллекции в качестве обучающих примеров

Следующий шаг — это собрать как можно больше обучающих примеров, чтобы обучить вашу модель машинного обучения. Причем в этом проекте для сбора обучающих примеров вам нужно будет всего лишь играть в игру! Забавно, правда?

5. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы, затем нажмите на кнопку «Создать» (рис. 12.10).

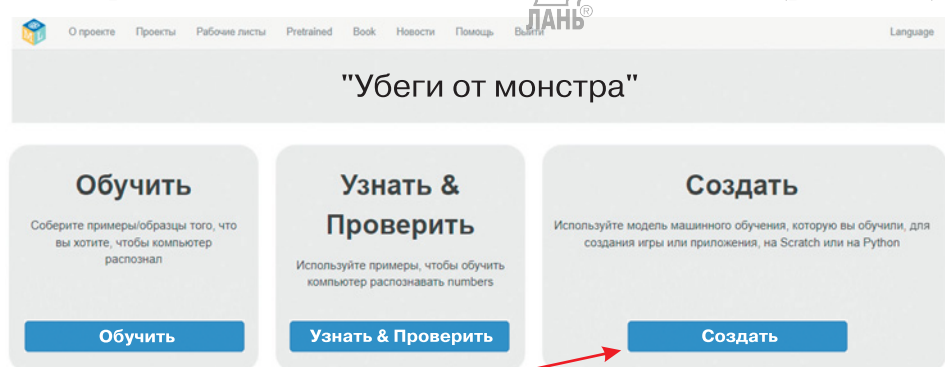


Рис. 12.10. Нажмите на кнопку «Создать», чтобы перейти к проекту Scratch

6. Нажмите на кнопку «Scratch 3» (рис. 12.11).

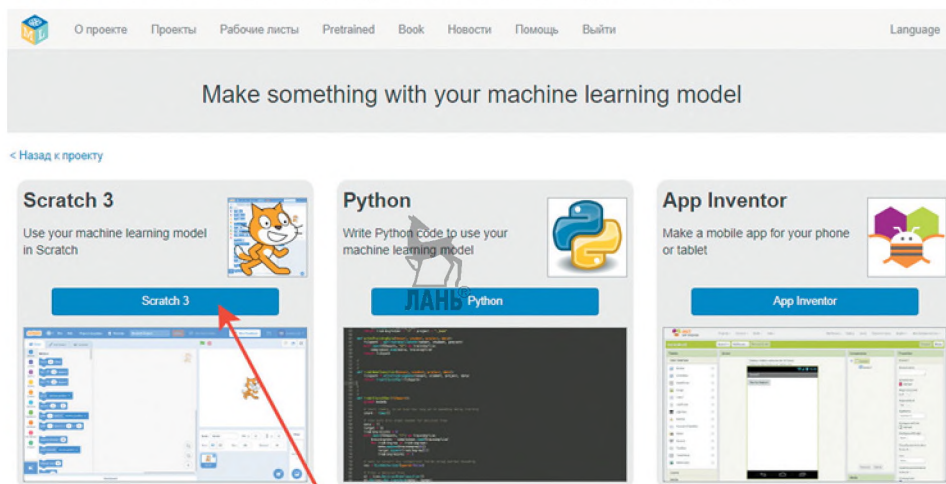


Рис. 12.11. Нажмите на кнопку «Scratch 3», чтобы вернуться к проекту Scratch

7. Нажмите на кнопку «straight into Scratch» (англ.: перейти сразу в Scratch), как показано на рисунке 12.12.



Примечание

На этой странице вы увидите уведомление, что в вашем проекте еще нет модели машинного обучения. Но ничего страшного: вы обучите ее позже, а сейчас вам нужны обучающие примеры, поэтому смело переходите прямо в Scratch.

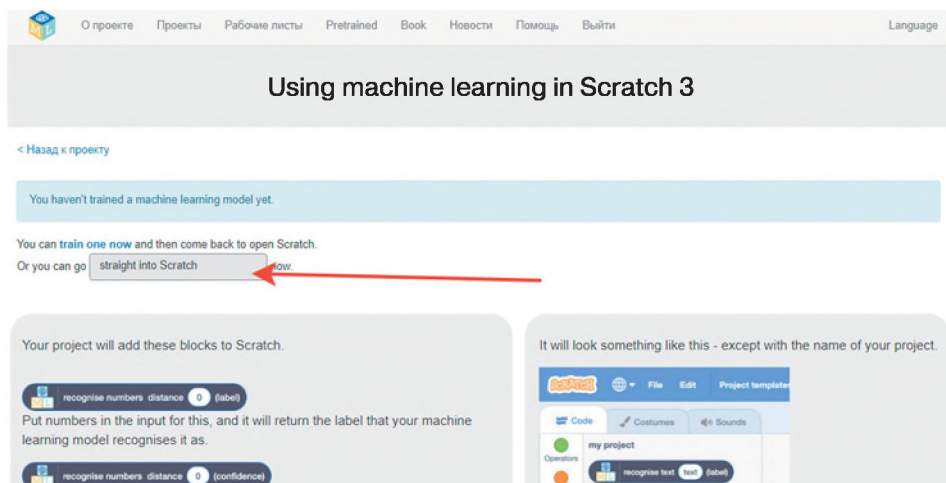


Рис. 12.12. Нажмите на кнопку «straight into Scratch», даже если в вашем проекте еще нет модели машинного обучения

8. Выберите в главном меню пункт **Шаблоны проектов**.

9. В открывшемся списке доступных шаблонов снова выберите проект с именем «Avoid the monster» (англ.: убеги от монстра). Но на этот раз в проекте появится категория блоков с именем вашего проекта машинного обучения.

10. Нажмите на вкладку «Сцена» в правом нижнем углу экрана. В Области кода найдите первый фрагмент скрипта «Когда нажат зеленый флажок», как показано на рисунке 12.13.

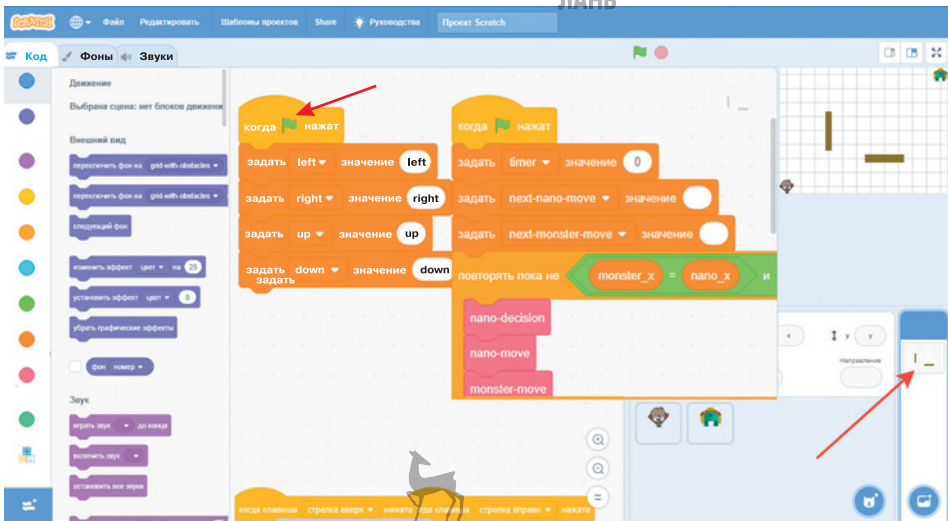


Рис. 12.13. Найдите первый фрагмент скрипта «Когда нажат зеленый флажок» (тот, что короче)

11. На Панели инструментов выберите категорию блоков с именем вашего проекта машинного обучения. Перетащите блоки с именами ваших коллекций в скрипт так, как показано на рисунке 12.14. Убедитесь, что названия коллекций совпадают с направлениями. Например, блок с названием коллекции «go_left» нужно перетащить в блок «задать left значение...».

12. Прокрутите вниз Область кода и найдите фрагмент скрипта «определить nano-decision» (рис. 12.15).

13. На Панели инструментов выберите категорию с именем вашего проекта. Перетащите из нее в скрипт блок «add training data» (англ.: добавить обучающие примеры), как показано на рисунке 12.16. Этот блок будет сохранять результаты каждого хода в соответствующие коллекции обучающих примеров. Выберите категорию «Переменные» на Панели инструментов и перетащите в блок «add training data» блоки с именами переменных,



Рис. 12.14. Добавьте в скрипт блоки с названиями ваших коллекций обучающих примеров

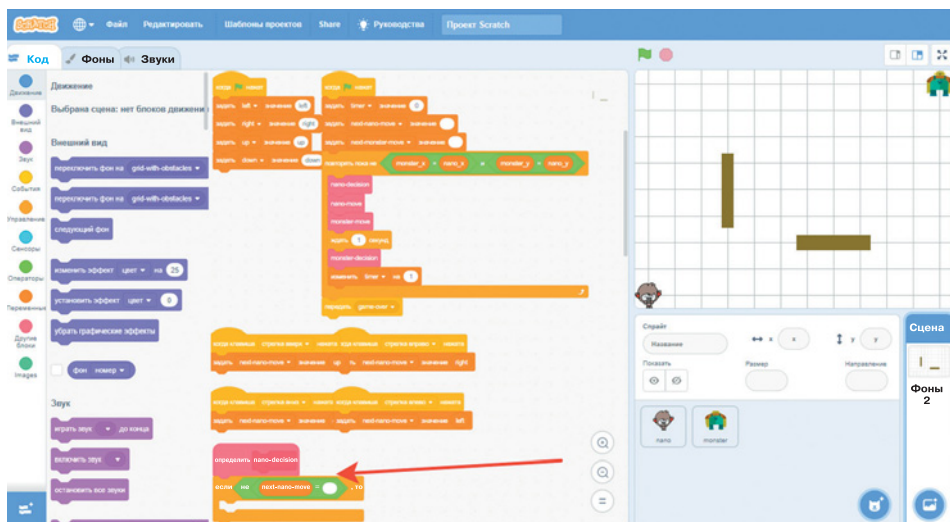


Рис. 12.15. Найдите фрагмент скрипта «определить papo-decision»

которые показаны на рисунке 12.16. Теперь для каждого хода будут сохраняться координаты персонажей папо и монстра вместе с решением, которое вы приняли (куда двигаться дальше). Так вы и будете собирать обучающие примеры для своей модели машинного обучения.

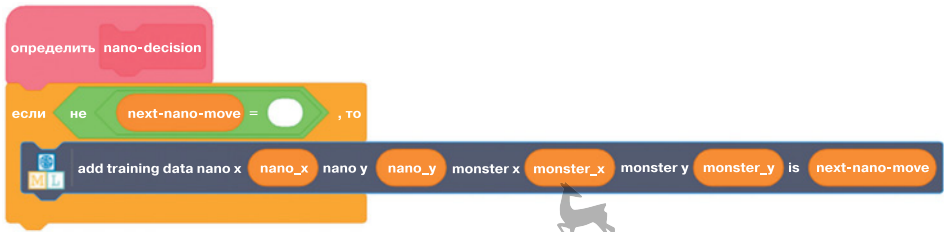


Рис. 12.16. Обновите фрагмент скрипта «определить nano-decision»

14. Теперь вам нужно обязательно сохранить ваш проект, чтобы никакие изменения не потерялись и вы смогли вернуться к нему позже. Это очень важно! Сначала найдите в Главном меню Scratch поле для ввода текста. Наберите в нем имя текущей версии проекта — «Игра „Убегите от монстра“ ОБУЧЕНИЕ», как показано на рисунке 12.17. Это имя будет вам напоминать, что эта версия проекта нужна именно для сбора обучающих примеров вашей модели машинного обучения.

Теперь выберите в Главном меню **Файл** → **Сохранить на свой компьютер**.

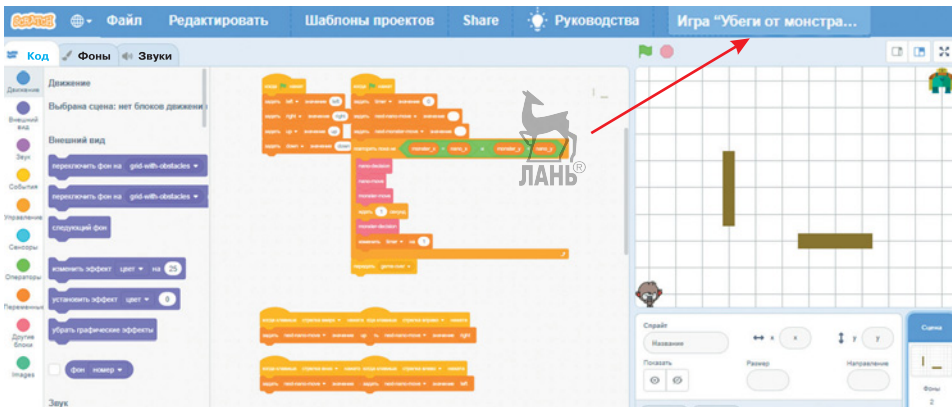


Рис. 12.17. Обновите имя проекта, перед тем как сохранить текущую версию

15. Настало время поиграть!

Нажмите на кнопку «Полноэкранный режим» в правом верхнем углу экрана, а затем — на зеленый флажок, чтобы запустить игру. Управляйте персонажем nano с помощью стрелок на клавиатуре, как вы делали раньше.

Постарайтесь убегать от монстра как можно дольше. Чем лучше вы будете играть, тем больше обучающих примеров получит ваша модель машинного обучения и тем лучше она сможет обучиться. Я в вас верю!

Когда вы решите, что играете уже достаточно долго, нажмите на красный знак «Стоп».

Теперь, если вернуться к проекту машинного обучения, то на этапе обучения вы увидите записи всех своих ходов в соответствующих коллекциях обучающих примеров (рис. 12.18). Если коллекции все еще пустые — обновите страницу, чтобы увидеть последние обновления.



Рис. 12.18. Все ходы, которые вы сделали во время игры, должны отобразиться в коллекциях обучающих примеров

16. Сыграйте в игру еще несколько раз, пока не соберете все возможные примеры ситуаций, в которых может оказаться папо.

17. Наконец-то настало время обучить вашу модель машинного обучения!

Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы, а затем на кнопку «Узнать и проверить» (рис. 12.19).

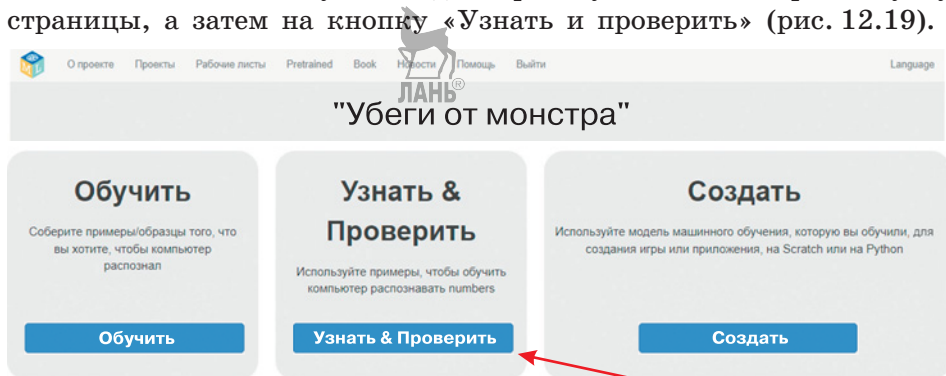


Рис. 12.19. Нажмите на кнопку «Узнать и проверить», чтобы обучить модель машинного обучения на примерах ходов, которые вы делали в игре

18. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 12.20).

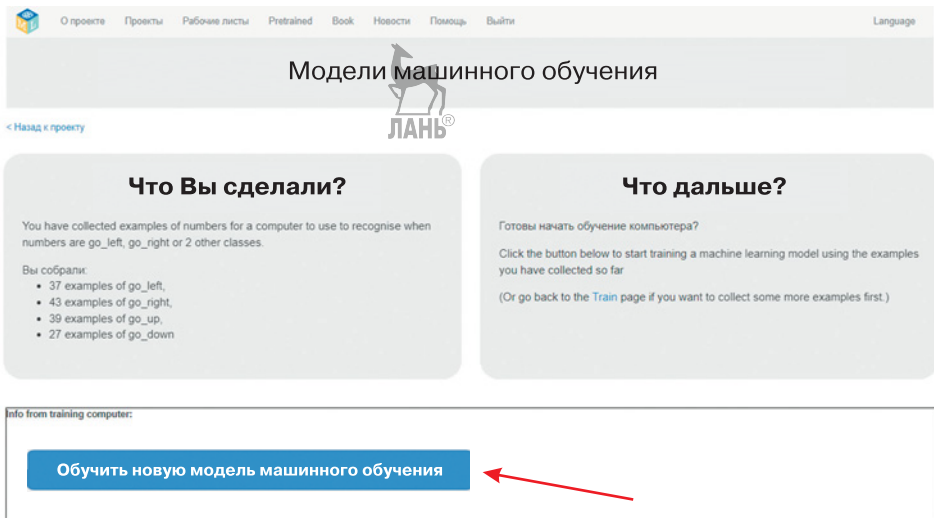


Рис. 12.20. Обучите модель машинного обучения на примерах ходов, которые вы делали в игре

Тестирование игры

Представляете, вы только что обучили свою модель машинного обучения играть в игры! Как здорово! Теперь нужно проверить, насколько хорошо она справляется. Лучший способ это сделать — дать ей возможность управлять персонажем папо и посмотреть, как долго он сможет убежать от монстра.

Для этого вам нужно будет изменить свой проект Scratch так, чтобы он управлялся моделью машинного обучения, а не стрелками на клавиатуре.

1. В Области кода найдите фрагменты скрипта «когда клавиша ... нажата». Таких фрагментов должно быть четыре (они показаны на рисунке 12.21). Это фрагменты скрипта «когда клавиша „стрелка вверх“ нажата», «когда клавиша „стрелка вниз“ нажата», «когда клавиша „стрелка влево“ нажата» и «когда клавиша „стрелка вправо“ нажата».

Удалите эти фрагменты скрипта, они вам больше не понадобятся, ведь теперь игра будет управляться моделью машинного обучения. Чтобы это сделать, нажмите по очереди на каждый

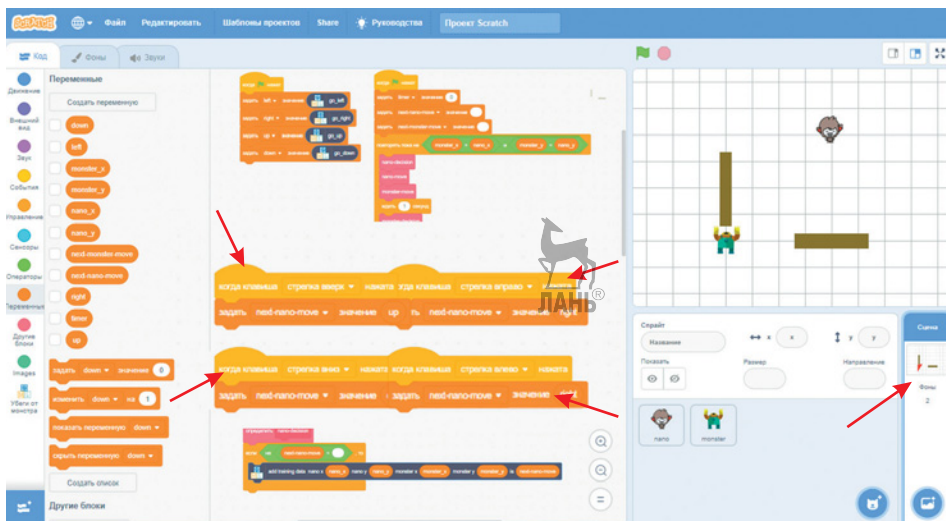


Рис. 12.21. Удалите 4 фрагмента скрипта «когда клавиша ... нажата», чтобы игрой больше нельзя было управлять с помощью клавиш со стрелками

из блоков, а затем на клавишу **Delete** (англ.: удалить) или щелкните правой кнопкой мыши и выберите пункт «Удалить блок». Убедитесь, что вы удалили все четыре фрагмента скрипта. Теперь персонаж nano не может управляться ни одной из клавиш со стрелками.

2. Найдите фрагмент скрипта «определить nano-decision», который вы недавно обновляли. Используя блоки из категории «Переменные» и категории с именем вашего проекта машинного обучения, обновите скрипт так, как показано на рисунке 12.22.



Рис. 12.22. Обновите фрагмент скрипта «определить nano-decision», чтобы теперь ваша модель машинного обучения могла управлять игрой

Теперь компьютер не будет обучаться на ваших ходах, а будет самостоятельно принимать решения (используя модель машинного обучения).

3. Найдите фрагмент скрипта «Когда нажат зеленый флажок» (второй, тот, что длиннее). Удалите из него блок «ждать 1 секунду».

Теперь игра будет идти быстрее, и вам не придется ждать каждого нового хода. Обновленный скрипт показан на рисунке 12.23. Стрелка на рисунке указывает на то место, где находится блок, который нужно удалить.

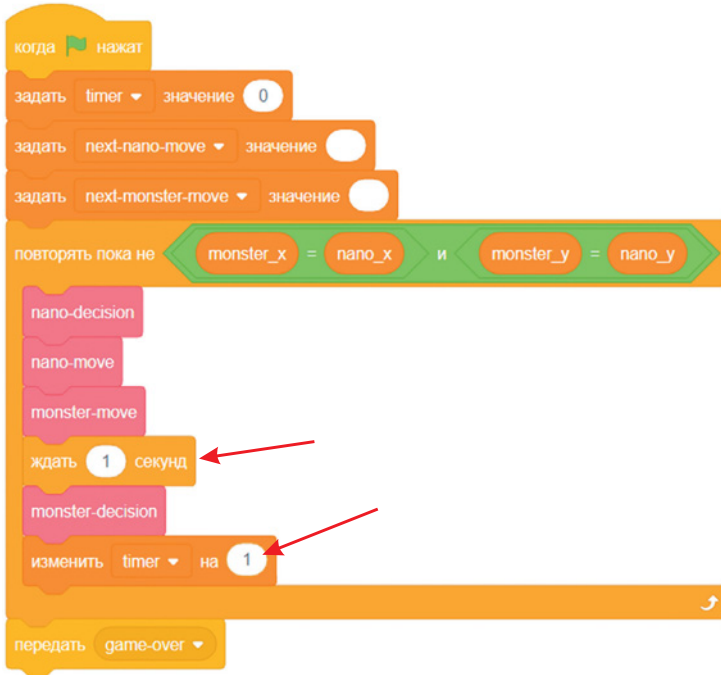


Рис. 12.23. Удалите блок «ждать 1 секунду»

4. Теперь вам нужно сохранить проект снова, чтобы иметь возможность вернуться к нему позже. Но сохраните его как другую версию, под именем «Игра „Убег от монстра“ ТЕСТИРОВАНИЕ». Так вы будете понимать, что в этой версии проекта игру ведет модель машинного обучения, а не вы, как в предыдущей сохраненной версии. Затем выберите пункт меню **Файл** → **Сохранить на свой компьютер**.

5. Нажмите на кнопку перехода в полноэкранный режим, а потом на зеленый флажок.

Теперь вам остается только наблюдать, как ваша модель машинного обучения пытается уберечь папо от монстра (рис. 12.24). Вы молодцы!

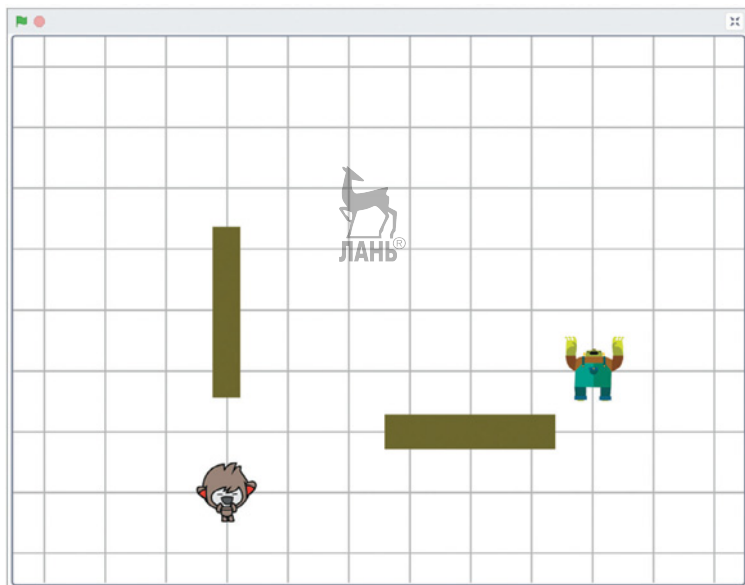


Рис. 12.24. Ваша модель машинного обучения в действии

Обзор и улучшение проекта

Интересно, насколько хорошо справилась ваша модель машинного обучения?

Чем дольше она может уберечь папо от монстра, тем лучше она обучилась. Если вы поработали действительно хорошо, когда собирали обучающие примеры, она может играть почти бесконечно. Потому что монстр движется с точно такой же скоростью, что и папо, а значит, если модель не ошибается в своих решениях, то папо может убегать от монстра вечно.

Но что же дает разница в количестве обучающих примеров?

Попробуйте снова открыть версию проекта Scratch, которая называется «...ОБУЧЕНИЕ». Поиграйте в игру еще какое-то время, чтобы собрать больше обучающих примеров для проекта машинного обучения. Затем перейдите к этапу «Узнать и проверить» и снова обучите свою модель машинного обучения. Наконец, откройте версию проекта Scratch, которая называется «...ТЕСТИРОВАНИЕ», и посмотрите, как ваша модель будет справляться с игрой теперь.

Помогло ли добавление новых обучающих примеров?

Сделайте это еще несколько раз и посмотрите, как объем обучения влияет на успехи модели.

Модель машинного обучения, которую вы создали в этом проекте, относится к типу моделей, который называют **классификатор дерева решений**, потому что алгоритм принятия решений такими моделями может быть изображен в виде своеобразного дерева, как показано на рисунке 12.25. Для того чтобы увидеть диаграмму дерева решений для вашей модели, вернитесь к этапу «Узнать и проверить» и нажмите на кнопку «Описать вашу модель» рядом с кнопкой тестирования^{*} (рис. 12.26).

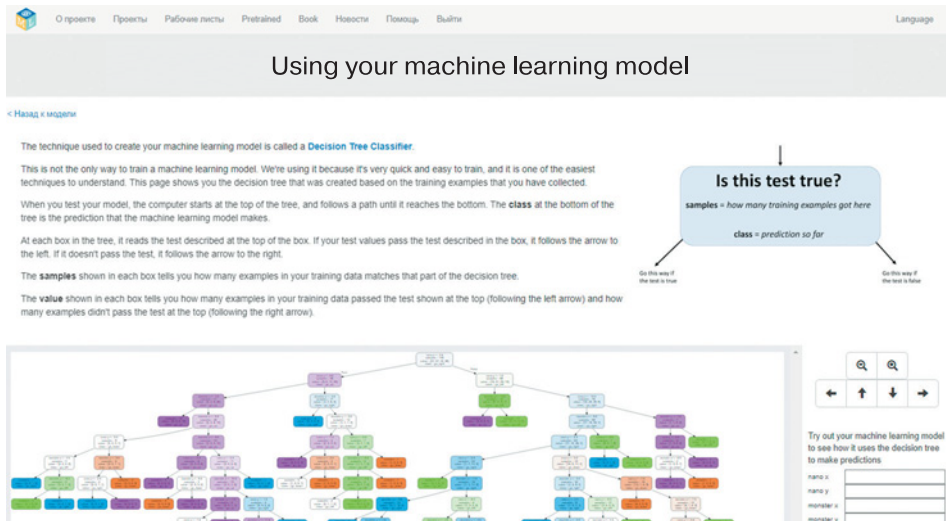


Рис. 12.25. Дерево решений

Диаграмма дерева решений помогает понять, как ваша модель машинного обучения *предсказывает* результаты каждого хода.

Каждая ячейка на дереве описывает тестовое условие. Например, «монстр $x < 3$ » означает «координата x монстра меньше 3?». Если условие верно, дерево направляет по левой стрелке. Если тест неверен, дерево велит двигаться по стрелке вправо.

Модель машинного обучения начинает обход с вершины дерева и следует стрелкам, определенным тестовыми условиями, пока не достигнет нижней части дерева (одного из концов).

Чтобы наглядно увидеть, как модель принимает решения, введите некоторые координаты папо и монстра в поля справа от вашего дерева решений и нажмите на кнопку «Проверить». На диаграмме дерева решений^{*} будет показано, какой прогноз

^{*} Такие размышления называются «дерево принятия решений». Они широко используются в машинном обучении и анализе данных. — Прим. ред.

Модели машинного обучения



< Назад к проекту

Что Вы сделали?

You have trained a machine learning model to recognise when numbers are go_left, go_right or 2 other classes.

You created the model on Friday, September 23, 2022 10:37 PM.

You have collected:

- 37 examples of go_left,
- 43 examples of go_right,
- 39 examples of go_up,
- 27 examples of go_down

Что дальше?

Try testing the machine learning model below. Enter an example of numbers below, that you didn't include in the examples you used to train it. It will tell you what it recognises it as, and how confident it is in that.

If the computer seems to have learned to recognise things correctly, then you can go to Scratch and use what the computer has learned to make a game!

If the computer is getting too many things wrong, you might want to go back to the [Train](#) page and collect some more examples

Once you've done that, click on the button below to train a new machine learning model and see what difference the extra examples will make!

Try putting in some numbers to see how it is recognised based on your training

nano x	
nano y	
monster x	
monster y	

Тест
Describe your model

Рис. 12.26. Нажмите на кнопку «Описать вашу модель»

сделала ваша модель машинного обучения для этих координат в виде пути решений. Когда стрелок и условий больше не будет и модель достигнет нижней части дерева, вы найдете окончательный прогноз — итоговый результат всех размышлений (рис. 12.27).

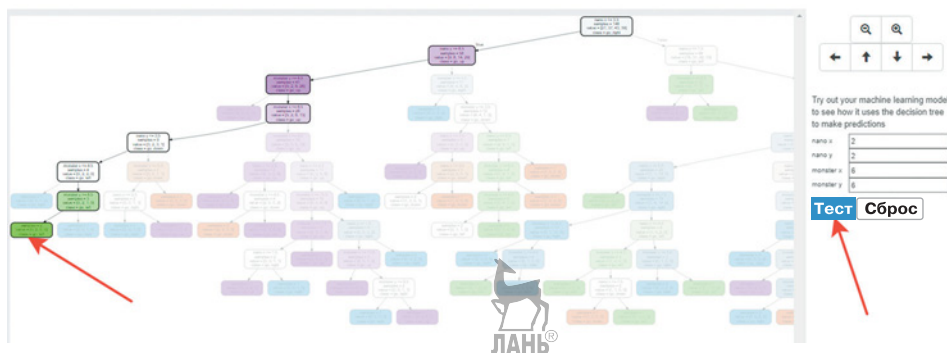


Рис. 12.27. На рисунке выделены координаты, используемые для предсказания результата хода

Протестируйте вашу модель на разных наборах координат, чтобы понять, насколько хорошо она научилась предсказывать ходы.

Что вы узнали

В этой главе вы узнали, что модели машинного обучения могут распознавать закономерности еще и в наборах чисел. Вы использовали классификатор дерева решений, чтобы обучить свою модель играть в упрощенную версию легендарной игры «Рас-Ман». Она использует наборы координат x и y для принятия решений о своих следующих шагах на основе их прогнозируемого результата.

Дерево решений не единственный способ обучить модель машинного обучения делать прогнозы на основе чисел. Но этот способ очень популярный, потому что его можно очень быстро реализовать и он один из самых простых для понимания. Например, в предыдущих главах вы использовали нейронные сети, которые могут быть более мощными, но более сложными и трудными для понимания.

Вы убедились, что, как и в случае с другими моделями машинного обучения, которые вы обучали ранее, эффективность модели увеличивается по мере того, как вы собираете больше обучающих примеров. В следующей главе вы узнаете больше о том, какую же роль играет количество обучающих примеров в проектах машинного обучения.





ГЛАВА 13

Создание игры «Крестики-нолики»

В предыдущей главе вы узнали, как компьютерные игры вроде «Рас-Ман» могут влиять на развитие машинного обучения. Но есть еще одна игра, которую вы наверняка знаете и которая используется для развития машинного обучения еще чаще и дольше. И это — легендарные «Крестики-нолики».

Свою вариацию игры предоставил Дональд Мичи, известный британский ученый, исследователь искусственного интеллекта. В 1960 году он разработал программу MENACE (Machine Educable Noughts and Crosses Engine, в переводе с англ.: машинообучаемая программа для игры в «Крестики-нолики», или, дословно, «Угроза»). Это была одна из первых программ, которая научилась играть в «Крестики-нолики» против человека почти идеально.

Основной принцип работы MENACE был продемонстрирован с помощью механического компьютера, состоящего из спичечных коробков и стеклянных бусин внутри их, которые нужно было брать согласно обучающемуся алгоритму (рис. 13.1). Кстати, это отличное напоминание нам с вами о том, что основные идеи, которые лежат в основе машинного обучения, развивались многие десятилетия.



Рис. 13.1. Воссоздание MENACE — машины Дональда Мичи для игры в «Крестики-нолики» (источник: Мэтью Скромгс, <https://commons.wikimedia.org/wiki/File:Mscroggs-MENACE-cropped.jpg>)

Тем не менее «Крестики-нолики» далеко не единственная игра, которая оказала большое влияние на развитие искусственного интеллекта. Еще один хороший пример — шахматы. В главе 1 я уже упоминал шахматный суперкомпьютер Deep Blue от компании IBM, который обыграл чемпиона мира по шахматам Гарри Каспарова. Но это случилось только спустя несколько десятилетий с момента начала работы над созданием компьютера, умеющего играть в шахматы. Еще в 1950-х годах известный математик Алан Тьюринг написал статью под названием «Цифровые компьютеры в применении к играм», в которой он задавался вопросами: можно ли создать машину, которая будет играть в шахматы, и можно ли заставить ее обучаться на собственном опыте?

В последние годы исследовательское сообщество в области искусственного интеллекта особое внимание уделяет более сложным играм, чем «Крестики-нолики» и даже шахматы. Например, настольной игре «Го». Чтобы выиграть в «Го», невозможно полагаться на компьютерный подход «грубой силы», такой как у Deep Blue (который исследует все возможные ходы и позиции) — из-за огромного количества возможных ходов и стратегий. Компьютер DeepMind AlphaGo от компании Google совершил прорыв в исследованиях в области искусственного интеллекта в 2016 году. Он победил чемпиона мира по «Го» Ли Седоля.

Инструменты для создания нейронных сетей становятся все проще в использовании, а наши компьютеры — все быстрее и мощнее. Их возможности уже выходят за рамки учебных и исследовательских задач в области искусственного интеллекта. Например, если вы введете поисковый запрос «нейросеть „Супер Марио“», то найдете огромное количество примеров и руководств по обучению нейросети играть в «Крестики-нолики» (рис. 13.2).

Теперь вы знаете, что все это возможно. Но вы еще только учитесь, поэтому давайте все же начнем с чего-то достаточно простого и понятного. В проекте этой главы вы воспроизведете упрощенную версию MENACE в Scratch и обучите свою модель машинного обучения играть в «Крестики-нолики» (рис. 13.2).

Давайте же скорее начнем! Будет интересно!



Рис. 13.2. «Крестики-нолики» — это отличная игра для исследований в области машинного обучения!

Выполнение проекта

Вы наверняка уже знакомы с игрой «Крестики-нолики» и играли в нее сотни раз. Но, на всякий случай, давайте сначала вспомним, как это делается (с помощью проекта Scratch), а потом вместе подумаем, как можно связать эту игру с машинным обучением.

Перейдите по ссылке <https://machinelearningforkids.co.uk/scratch3/> и в открывшемся окне Scratch выберите пункт Главного меню **Шаблоны проектов** (рис. 13.3). В списке доступных шаблонов найдите проект с именем «Noughts and Crosses» (англ.: крестики и нолики). Этот шаблон представляет собой простой вариант игры «Крестики-нолики» в Scratch. Нажмите на зеленый флажок, чтобы запустить игру.

Вы будете ставить на игровом поле крестики (X), а компьютер — нолики (O). Сейчас компьютер использует не слишком «умную» стратегию игры, но это сделано преднамеренно. Ведь ваша задача в этом проекте — улучшить ее.

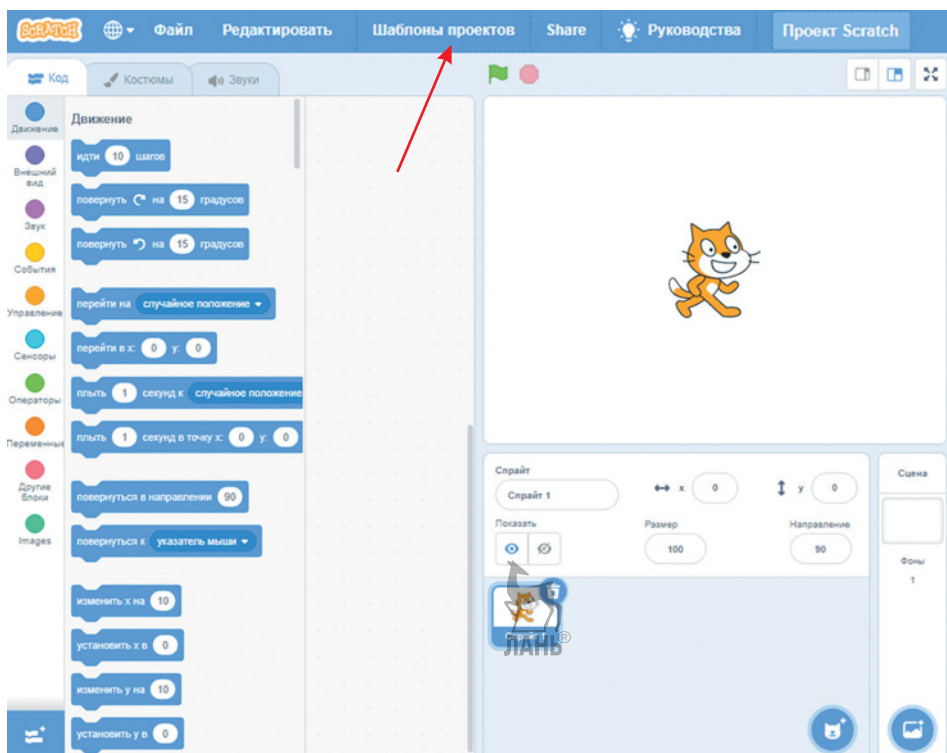


Рис. 13.3. Пункт **Шаблоны проектов** в Главном меню Scratch

Попробуйте поиграть в эту игру и понять, какими правилами руководствуется компьютер, когда делает свои шаги. А после этого, чтобы проверить свои догадки, перейдите в Область кода — в ней описана вся логика поведения компьютера в игре.

Теперь давайте приступать к выполнению проекта. Первый вопрос: как изобразить игровое поле? Есть много способов это сделать, но мы с вами начнем с самого простого — разделим игровое поле на 9 ячеек и пронумеруем каждую из них так, так показано на рисунке 13.4.

В шаблоне «Noughts and Crosses» игровое поле представлено именно так, и вы можете видеть номера ячеек в скрипте (рис. 13.5).

1	2	3
4	5	6
7	8	9

Рис. 13.4. Один из возможных способов реализовать игровое поле — разделить его на ячейки и пронумеровать их от 1 до 9

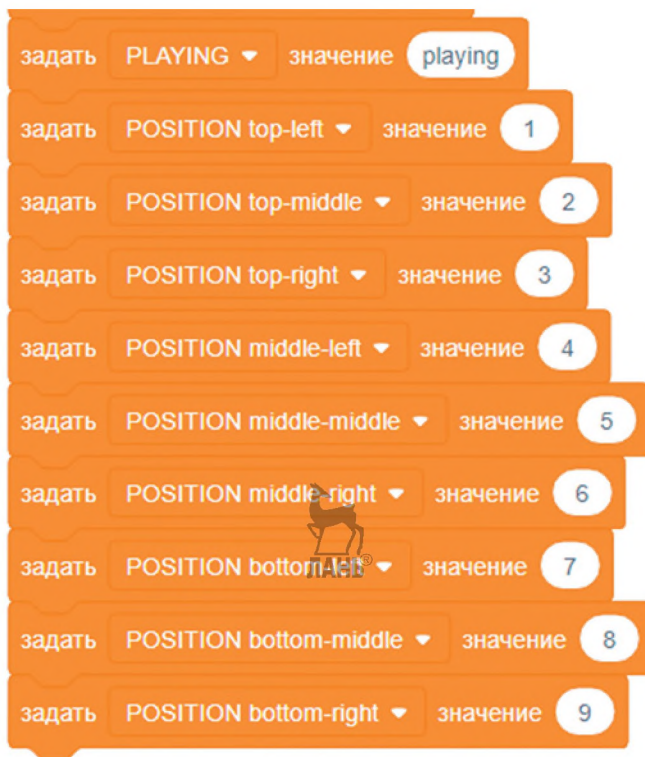


Рис. 13.5. Представление игрового поля в виде пронумерованных ячеек в шаблоне «Noughts and Crosses»

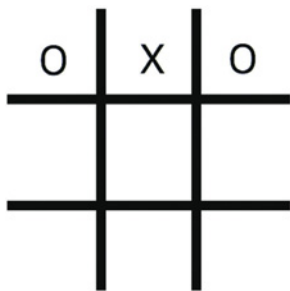


Рис. 13.6. Пример одного из состояний игры

Вам также нужно будет описывать позиции крестиков и ноликов на поле. Чтобы создать действительно хорошую модель машинного обучения, вам нужно будет обучать ее на примерах стратегий победителей (неважно, кто именно выиграл, крестики или нолики). Но чтобы не запутаться, используйте наименования «игрок» для сохранения выигрышных ходов и «соперник» для описания проигрышных ходов. Например, представьте, что игра, которую выиграли крестики (X), начиналась так, как показано на рисунке 13.6.

Это состояние игрового поля можно описать следующим образом:

Левая ячейка в верхнем ряду	Соперник
Средняя ячейка в верхнем ряду	Игрок
Правая ячейка в верхнем ряду	Соперник
Левая ячейка в среднем ряду	Пусто
Средняя ячейка в среднем ряду	Пусто
Правая ячейка в среднем ряду	Пусто
Левая ячейка в нижнем ряду	Пусто
Средняя ячейка в нижнем ряду	Пусто
Правая ячейка в нижнем ряду	Пусто

Вам нужно обучить свою модель машинного обучения таким образом, чтобы, исходя из текущего состояния игрового поля, она могла определять, какой следующий ход нужно сделать. Чтобы это сделать, вам понадобятся обучающие примеры — примеры ходов, которые приведут ситуацию к выигрышу. Каждый пример должен обязательно содержать следующую информацию:

- как выглядело игровое поле до текущего хода;
- какой ход был сделан.

При этом в качестве обучающих примеров будут сохранены только ходы того игрока, который выиграл игру. Другими словами, если вы, играя крестиками, выиграли игру, то вам нужно, чтобы в обучающие примеры попали все ходы крестиков (X). Иначе, если игру выиграл компьютер, в обучающие примеры должны попасть все ходы ноликов (O).

Подготовка проекта-игры

В предыдущем проекте вы собирали обучающие примеры, играя в игру «Рас-Мат». Это отличный способ — собирать обучающие примеры, не набирая их вручную, а играя, поэтому в этот раз вам предстоит начать с того же.

Но первый шаг — это, как всегда, подготовка коллекций. Потому что обучающие примеры нужно куда-то сохранять. Для этого проекта вам понадобится девять коллекций — по одной коллекции для каждой ячейки игрового поля. То есть для каждого места, куда можно сделать ход, в игре нужна будет своя коллекция.

1. Перейдите по ссылке <https://machinelearningforkids.co.uk/> Создайте новый проект машинного обучения и назовите его «Игра „Крестики-нолики“». В качестве распознаваемого типа данных выберите «числа».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Add a value» (англ.: добавить значение), в качестве имени значения укажите «Top Left» (англ.: левая верхняя ячейка). В выпадающем списке «Тип значения» выберите пункт «множественный выбор». В пункте «Варианты:» добавьте три варианта, назовите их «ПУСТО», «ИГРОК» и «СОПЕРНИК». Затем нажмите на кнопку «Add another value» (англ.: добавить другое значение) и добавьте таким образом еще восемь значений с типом значения «множественный выбор». Каждое из этих значений должно иметь те же три варианта: «ПУСТО», «ИГРОК» и «СОПЕРНИК», а сами значения, как вы уже наверняка догадались, должны называться:

- Top Middle (англ.: средняя верхняя ячейка)
- Top Right (англ.: правая верхняя ячейка)
- Middle Left (англ.: левая ячейка в среднем ряду)
- Middle Middle (англ.: средняя ячейка в среднем ряду)
- Middle Right (англ.: правая ячейка в среднем ряду)
- Bottom Left (англ.: левая нижняя ячейка)
- Bottom Middle (англ.: средняя нижняя ячейка)
- Bottom Right (англ.: правая нижняя ячейка)

Пожалуйста, обратите внимание, что все три варианта должны быть названы одинаково («ПУСТО», «ИГРОК» и «СОПЕРНИК») для каждого из девяти значений. Только в этом случае компьютер сможет понять, что для каждой ячейки на игровом поле до-

ступны одинаковые варианты развития событий. Если же вдруг вы наберете один из вариантов неправильно, вы всегда можете его удалить, нажав на красный крестик рядом с этим вариантом, а затем добавить правильный.

Когда вы закончите, страница должна выглядеть так, как показано на рисунке 13.7.

Рис. 13.7. Подготовка проекта

3. Нажмите на кнопку «Создать», чтобы создать проект и перейти к нему.

4. Нажмите на кнопку «Обучить», как показано на рисунке 13.8.

5. Нажмите на кнопку «Добавить новую метку» (рис. 13.9) и создайте девять коллекций, которые будут собирать обучающие примеры для каждой из девяти возможных ячеек на игровом поле. Назовите их «top left» (англ.: левая верхняя ячейка), «top middle» (англ.: средняя верхняя ячейка), «top right» (англ.: правая верхняя ячейка), «middle left» (англ.: левая ячейка в среднем ряду), «middle middle» (англ.: средняя ячейка в среднем ряду), «middle right» (англ.: правая ячейка в среднем ряду), «bottom left» (англ.: левая нижняя ячейка), «bottom middle» (англ.: средняя нижняя ячейка) и «bottom right» (англ.: правая нижняя

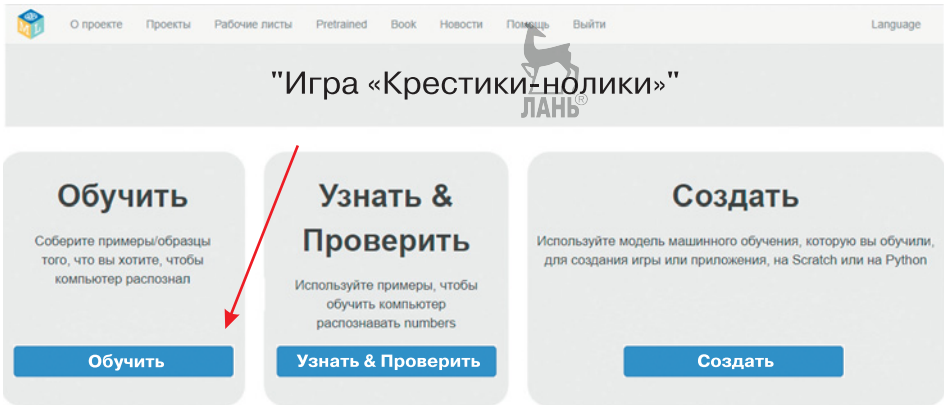


Рис. 13.8. Нажмите на кнопку «Обучить», чтобы начать подготовку коллекций обучающих примеров

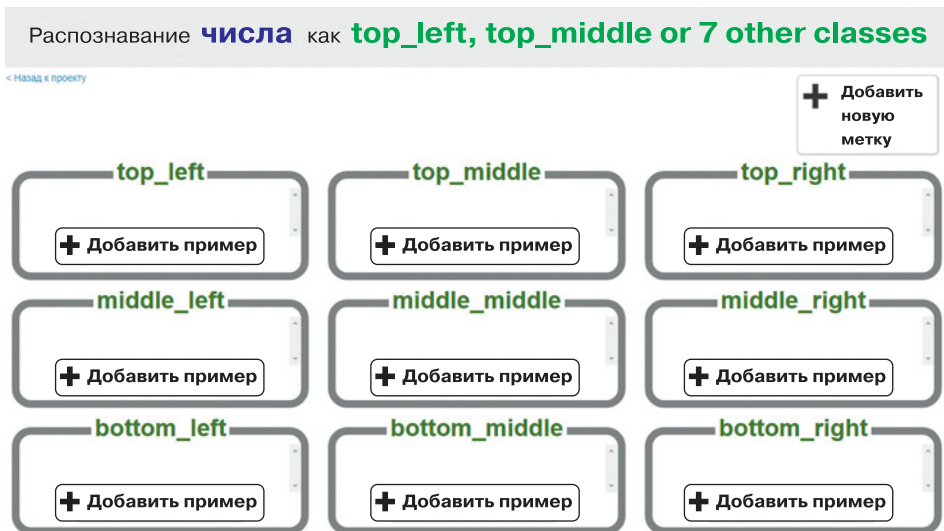


Рис. 13.9. Коллекции обучающих примеров для проекта «Игра „Крестики-нолики“»

ячейка). Символы «нижнее подчеркивание» между словами в названии коллекций будут добавлены автоматически.

Эти коллекции вы будете наполнять обучающими примерами — примерами ходов из игры «Крестики-нолики». Например, вспомните состояние игры, которое было показано на рисунке 13.6. Если игрок сделает свой следующий ход крестиками (X) в среднюю ячейку в среднем ряду, то новое состояние игрового поля будет сохранено как обучающий пример так, как показано на рисунке 13.10.

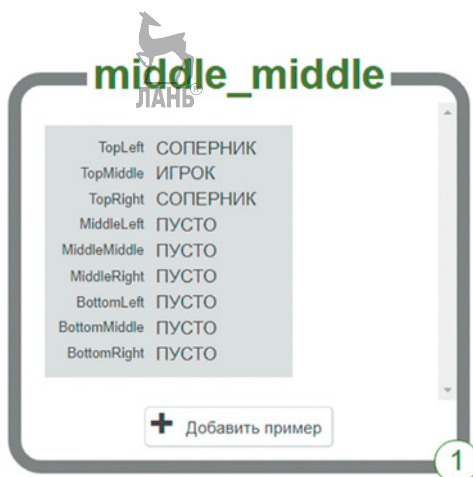


Рис. 13.10. Вот так будут записываться обучающие примеры для проекта «Игра “Крестики-нолики”»

Теперь вам нужно собрать много (действительно много) обучающих примеров для обучения своей модели машинного обучения.

6. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

7. Нажмите на кнопку «Создать».

8. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3» (рис. 13.11), чтобы открылось новое окно со средой Scratch.



Примечание

Вы увидите предупреждение о том, что у вас еще нет модели машинного обучения. Это нормально, так как в этом проекте для сбора обучающих примеров вы будете использовать Scratch.

9. Снова найдите и откройте проект с именем «Noughts and Crosses» (англ.: крестики и нолики) из списка шаблонов.

Скрипт выглядит точно так же, как когда вы открывали его в прошлый раз, но теперь на Панели инструментов появилась категория блоков с именем вашего проекта машинного обучения.

10. Нажмите на значок фона Сцены в правом нижнем углу экрана. В Области скриптов найдите фрагмент скрипта «setup model labels» (англ.: задать значения меткам модели), он показан на рисунке 13.12. Этот скрипт задает значения, которые будут использоваться в проекте.

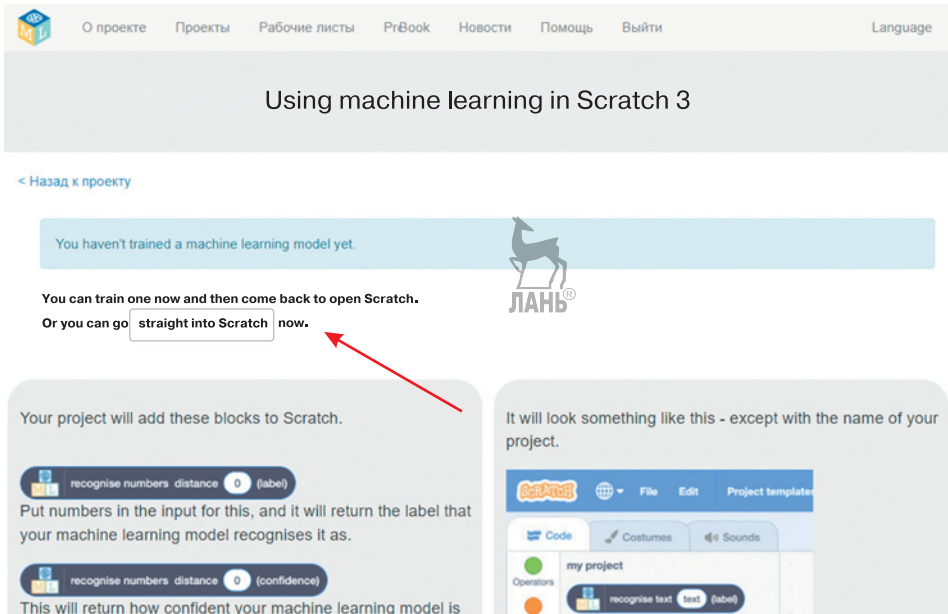


Рис. 13.11. Нажмите на кнопку «Открыть в Scratch 3» для перехода к Scratch, даже если у вас еще нет модели машинного обучения

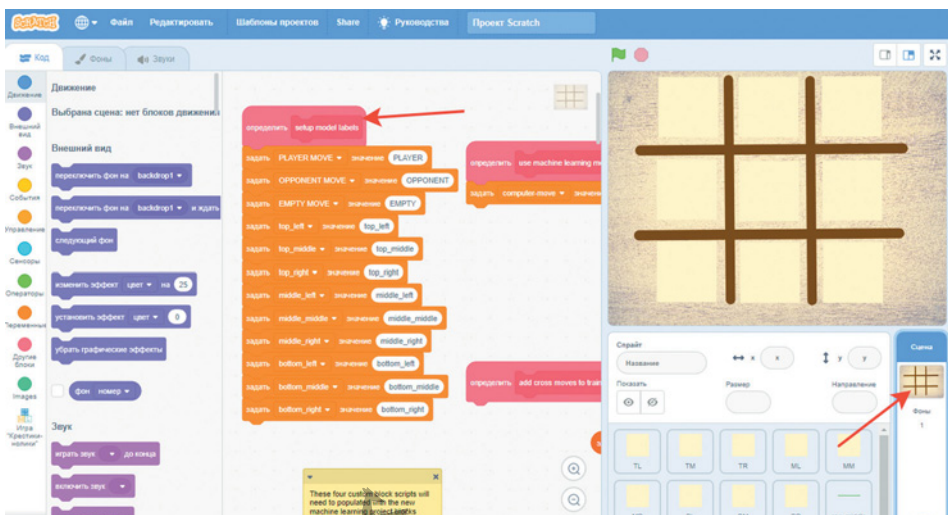


Рис. 13.12. Найдите фрагмент скрипта «setup model labels»

11. В Панели инструментов найдите категорию с именем вашего проекта машинного обучения. Перетащите из нее в скрипт «setup model labels» блоки с названиями коллекций так, как



Рис. 13.13. Дополните скрипт блоками из вашего проекта машинного обучения

это показано на рисунке 13.13. Делайте это очень внимательно, чтобы каждый из блоков оказался на своем месте, например, чтобы блок «top_left» оказался внутри блока «задать top_left значение...».

12. Проллистните Область кода вниз и найдите фрагменты скрипта «определить add cross moves to training data» и «определить add nought moves to training data» (рис. 13.14).

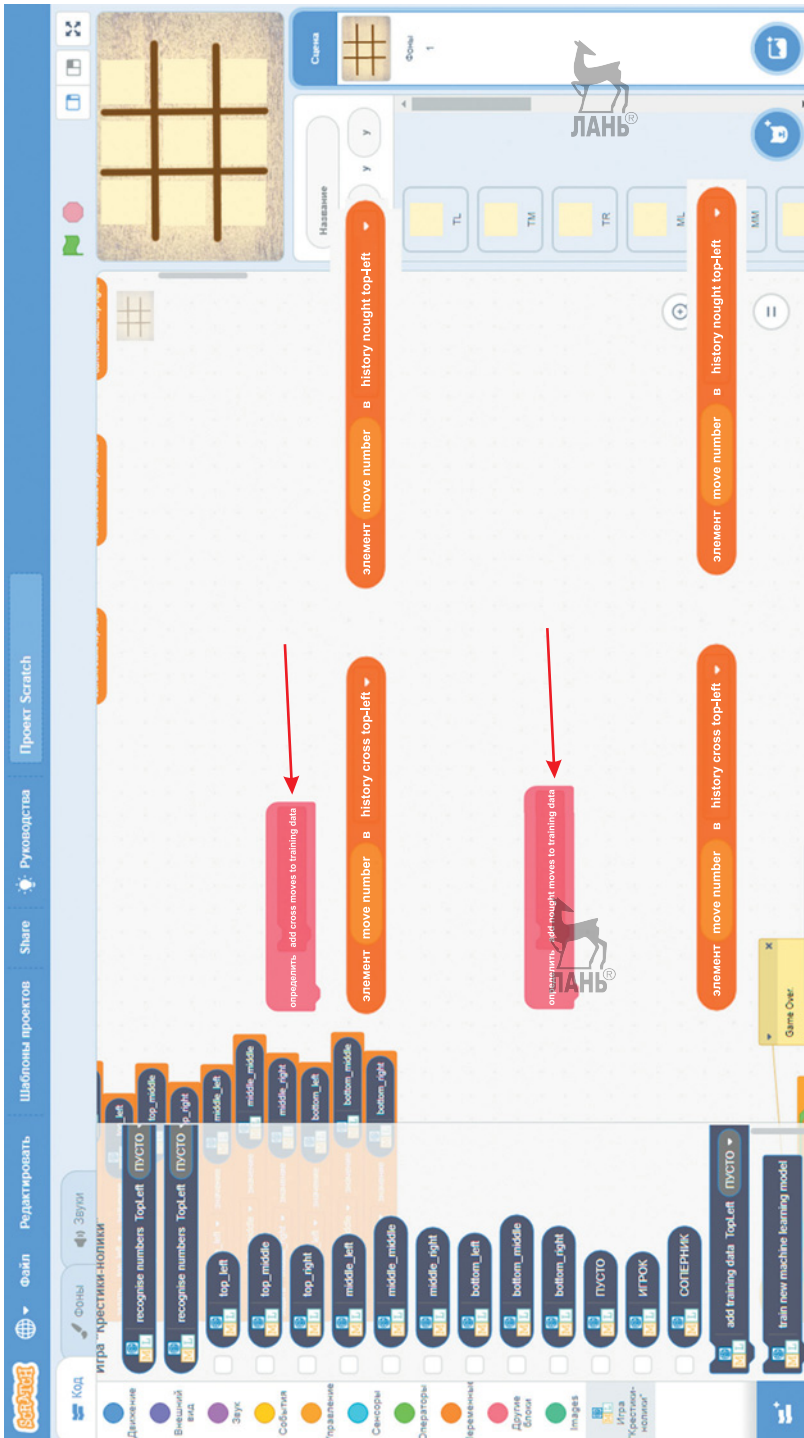


Рис. 13.14. Найдите фрагменты скрипта «определить...», которые вам предстоит настроить самостоятельно

13. В Панели инструментов найдите категорию с именем вашего проекта машинного обучения. Перетащите из нее в оба фрагмента скрипта «определить...» блок «add training data» (англ.: добавить обучающие примеры), как показано на рисунке 13.15.

14. Обновите блок «add training data» так, как показано на рисунке 13.16. На самом деле оранжевые блоки, которые вам нужны, уже есть в шаблоне проекта. Они должны находиться прямо под теми фрагментами скрипта, в которые их нужно добавить. Начиная слева, подтяните каждый блок к соединительному краю фрагментов скрипта над ними.

Еще раз внимательно проверьте все скрипты. Блоки, относящиеся к сохранению истории ходов крестиков (X), должны быть соединены с фрагментом скрипта «add cross moves to training data» (англ.: добавить ходы крестиков в обучающие примеры) именно так, как показано на рисунке 13.17.

Точно так же блоки, относящиеся к сохранению истории ходов ноликов (O), должны быть соединены с фрагментом скрипта «add nought moves to training data» (англ.: добавить ходы ноликов в обучающие примеры). Это очень важно!

А еще важно, чтобы названия блоков совпадали с названиями ячеек игрового поля. Например, ходы в среднюю ячейку в верхнем ряду должны сохраняться в коллекцию «TopMiddle» (рис. 13.18).

Убедитесь, что вы заполнили блоки для всех девяти ячеек игрового поля. Чтобы увидеть все нужные фрагменты скрипта, прокрутите Область кода вправо (рис. 13.19).

15. Теперь найдите фрагмент скрипта «когда я получу game over» (англ.: конец игры), который показан на рисунке 13.20. Этот фрагмент скрипта запускается для каждой игры и запускает на выполнение те самые скрипты добавления ходов крестиков и ноликов в коллекции обучающих примеров. Это те скрипты, которые вы только что дорабатывали.

А теперь добавьте в конец скрипта «когда я получу game over» блок «обучить новую модель машинного обучения», как показано на рисунке 13.21.

Это означает, что с этого момента после окончания каждой игры ходы победителя будут сохраняться и добавляться в коллекции в качестве обучающих примеров. Затем вы будете использовать этот обновленный набор обучающих примеров для обучения своей модели. Тем самым ваша модель машинного обучения будет становиться лучше и хитрее после каждой игры, в которую вы сыграли. Здорово, не правда ли?

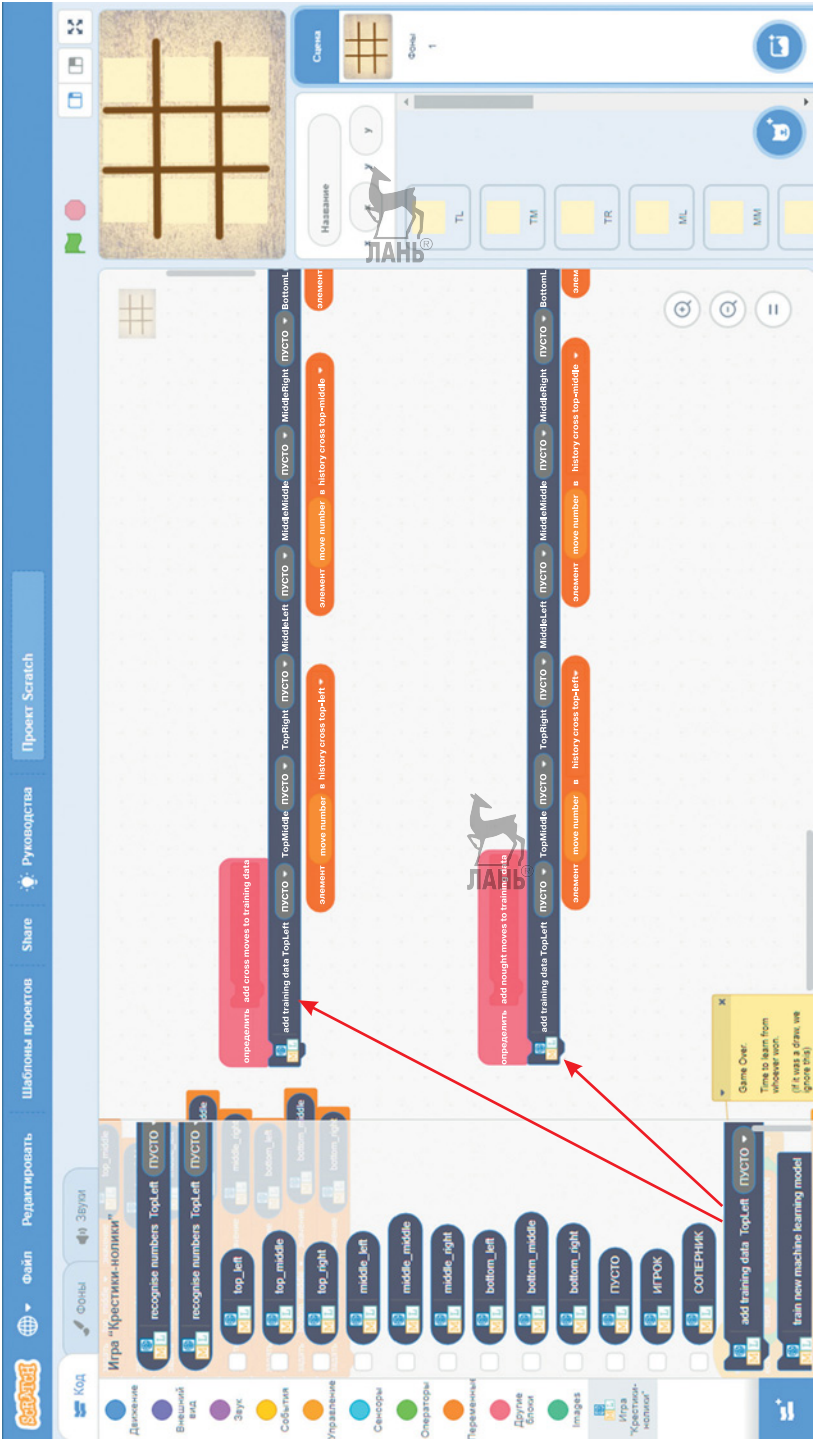


Рис. 13.15. Перетащите блок «add training data» в оба фрагмента скрипта «определить....»

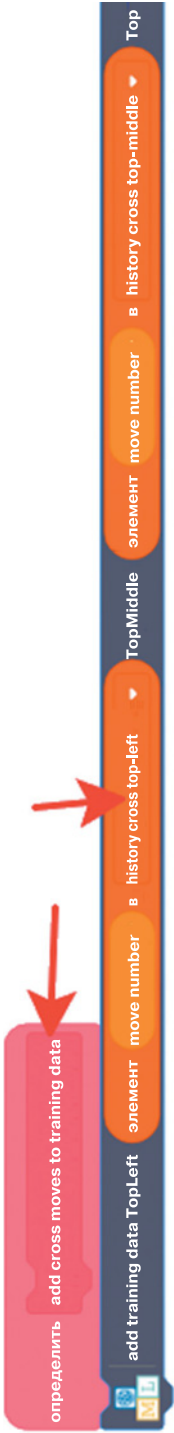


Рис. 13.17. Убедитесь, что вы добавили блоки, относящиеся к крестикам, в скрипт про крестики

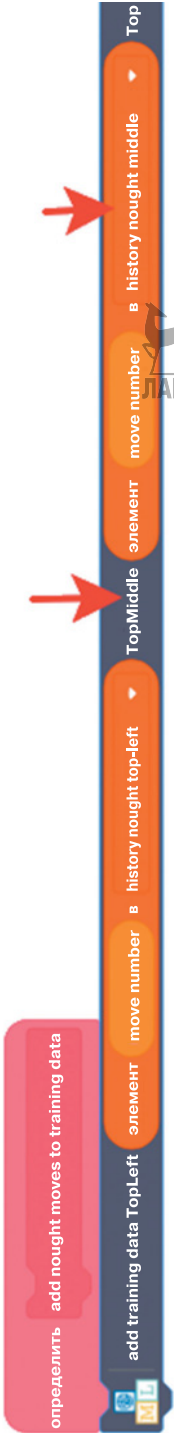


Рис. 13.18. Проверьте совпадение имен блоков с названиями ячеек игрового поля



Рис. 13.19. Убедитесь, что вы заполнили блоки для всех ячеек



Рис. 13.20. Найдите фрагмент скрипта «когда я получу game over»



Рис. 13.21. Теперь модель машинного обучения будет обучаться после окончания каждой игры

Обучение модели

Настало время поиграть!

1. Нажмите на кнопку «Полноэкранный режим» в правом верхнем углу экрана, а затем — на зеленый флажок, чтобы запустить игру. Сыграйте одну игру (рис. 13.22).

2. После того как игра закончится, вернитесь к проекту машинного обучения. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы, а затем на кнопку «Обучить». Вы увидите, что все ходы, которые сделал игрок-победитель, теперь отображаются в соответствующих коллекциях обучающих примеров, как на рисунке 13.23.



Рис. 13.22. Игра в «Крестики-нолики»

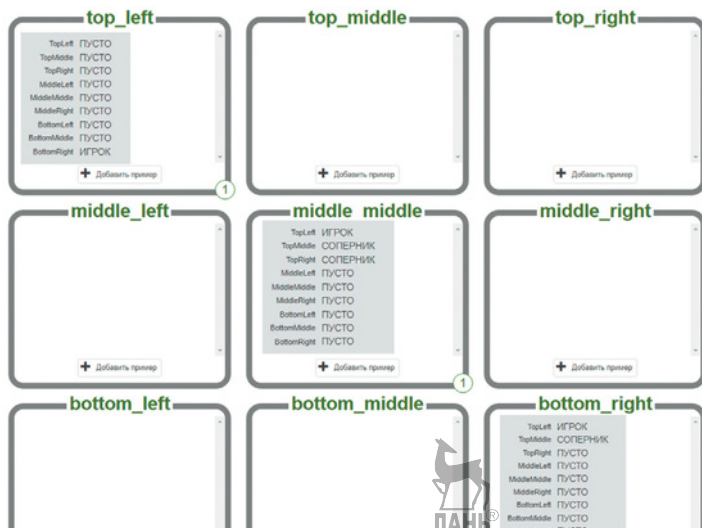


Рис. 13.23. Сравните полученные обучающие примеры с результатами игры на рисунке 13.22

3. Теперь, когда у вас есть обученная модель машинного обучения, настало время обновить проект Scratch для того, чтобы компьютер мог сам принимать решения и делать следующий ход. Прокрутите Область кода и найдите фрагмент скрипта «определить use machine learning model» (англ.: использование модели машинного обучения), который показан на рисунке 13.24.

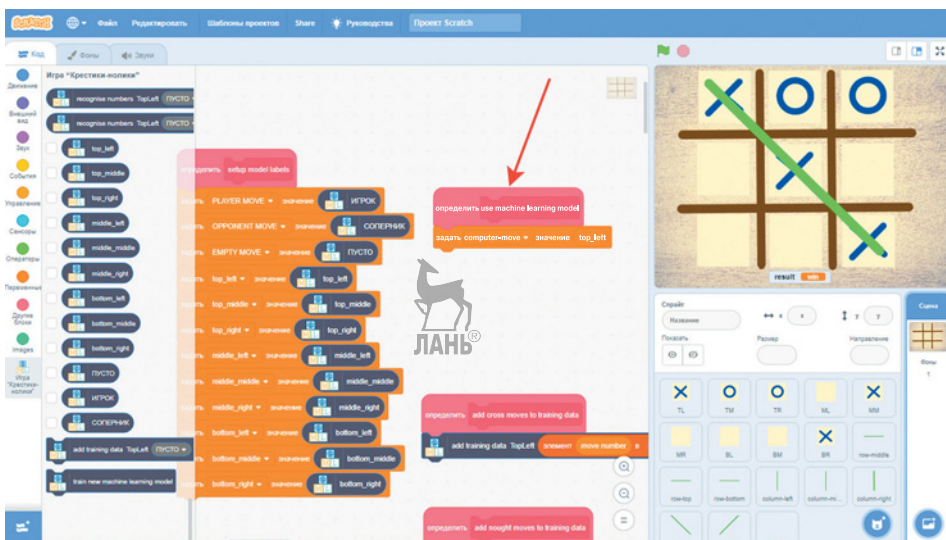


Рис. 13.24. Найдите фрагмент скрипта «определить use machine learning model»

Обновите этот скрипт так, как показано на рисунке 13.25, чтобы использовать вашу модель машинного обучения для принятия решения — какой ход лучше сделать.

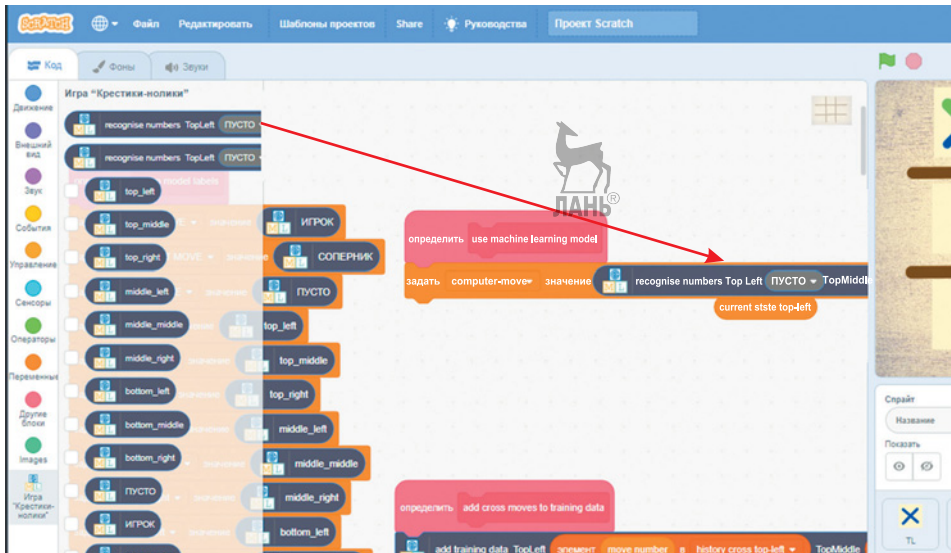


Рис. 13.25. Добавление блока «recognise numbers...» (англ.: распознать числа) в скрипт



Примечание

Убедитесь, что вы используете блоки «recognise numbers (название)», а не «recognise numbers (дать оценку)». Ведь вам нужно, чтобы модель анализировала сами данные, а не сравнивала цифры достоверности результатов.

4. Перетащите оранжевые блоки в эти блоки «recognise numbers (название)» так, как показано на рисунке 13.26. Как и раньше, эти блоки уже подготовлены для вас шаблоном проекта и даже находятся в нужных местах. Осталось только заполнить их и соединить. Получившиеся фрагменты скрипта передают вашей модели машинного обучения текущее состояние игрового поля, чтобы модель могла использовать эту информацию для определения лучшего следующего хода.



Рис. 13.26. Убедитесь, что названия блоков совпадают с именами ячеек игрового поля

Убедитесь, что вы заполнили эти блоки для всех девяти ячеек игрового поля и делали это слева направо. Также проверьте — все ли названия блоков и ячеек игрового поля совпадают? Например, блок «current state top-middle» (англ.: текущее состояние средней верхней ячейки) должен отправиться в коллекцию «TopMiddle».

Тестирование игры

Настало время протестировать ваш проект и узнать, как он работает!

Вы выполнили этот проект так, чтобы компьютер обучался, пока вы играете. А значит, вы должны увидеть, что с каждым разом компьютер играет все лучше и лучше. Но как это проверить?

Первый способ — это сыграть много игр и посчитать, сколько раз компьютер выиграл. А затем построить график и посмотреть, растет ли число побед компьютера, когда вы предоставляете ему больше обучающих примеров.

Я сыграл 300 игр в «Крестики-нолики» и подсчитал, сколько игр я выиграл, сколько проиграл и сколько сыграл вничью. Затем я изобразил результаты в виде гистограммы, которая показана на рисунке 13.27. Каждый столбик на гистограмме представляет собой 10 игр. Зеленым цветом обозначены игры, которые я выиграл. Оранжевый — игры, которые закончились ничьей. Красный — игры, которые выиграла модель машинного обучения.

Самый первый (крайний левый) столбик обозначает первые 10 игр. Как видите, я выиграл все 10.

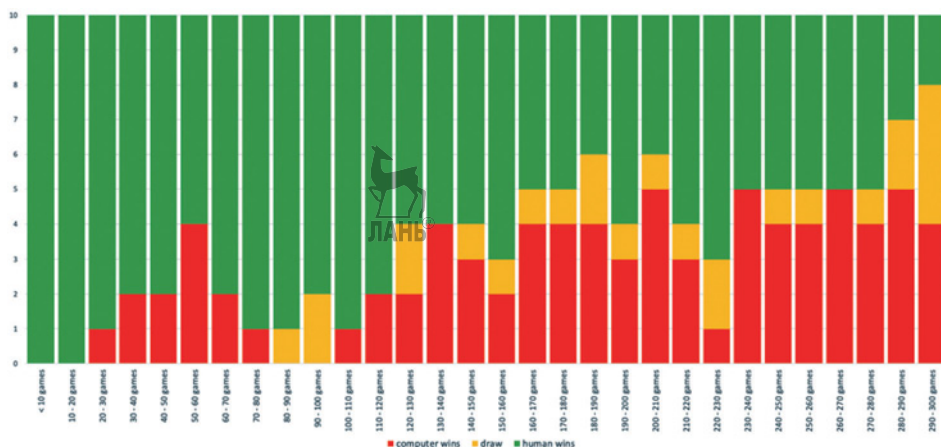


Рис. 13.27. Результаты игр в «Крестики-нолики», которые получил я

Следующий столбик обозначает следующие 10 игр. Здесь я снова выиграл все 10.

Последний же (крайний правый) столбик обозначает последние 10 игр. Я выиграл две из них, четыре закончились ничьей, и последние четыре я проиграл.

Что же получается? Свои первые 10 игр я выиграл с легкостью. Компьютер делал ходы «с закрытыми глазами», поэтому мне не составляло труда выиграть. Но когда я играл последние 10 игр, выиграть было не так уж легко. Мне нужно было быть очень и очень внимательным, стараясь не допускать ни единой ошибки. График, конечно же, не способен передать моих ощущений, но я действительно чувствовал, что компьютер становится умнее с каждой игрой.

Однако каждый проект машинного обучения может вести себя немного по-своему. Поэтому попробуйте обучить свою собственную модель и посмотреть, как ваш проект учится и совершенствуется. Надеюсь, вы поймете, что, чем больше обучающих примеров вы соберете, тем лучше будет работать ваша модель машинного обучения. И я также уверен, что вы почти наверняка обратите внимание на некоторые изменения в модели с течением времени, так же как и я в своем примере.

Обзор и улучшение проекта

Вы создали модель машинного обучения, которая учится играть в «Крестики-нолики», играя с вами. Однако самая большая проблема при самостоятельном обучении модели — это время, необходимое для того, чтобы сыграть сотни обучающих игр. Как вы думаете, есть ли более эффективные способы собрать много обучающих примеров?

Один из распространенных способов — привлечь к этому больше людей, чтобы они могли вам помочь. Представьте, что вместо того, чтобы мне самому играть в 300 игр, я сохранил свой проект Scratch, дал файл проекта 30 друзьям и попросил каждого сыграть по 10 игр. Разделение такой работы значительно облегчило бы обучение модели, так как сыграть всего по 10 игр не составило бы труда никому.

А теперь представьте, что можно попросить помочь не 30 человек, а 300. Или даже 3000!

Надеюсь, вы осознаете преимущества обучения модель машинного обучения с помощью большого количества людей. Такой

способ еще называют *краудсорсинг*. Тем не менее этому способу тоже присущи некоторые недостатки, такие как большие сложности с тем, чтобы найти достаточно многочисленную группу людей, координировать их, объяснять, чего вы хотите, всем им одновременно, следить за тем, чтобы они все делали так, как вы хотите, и так далее. Но даже с учетом этих недостатков привлечение большого количества людей для обучения моделей машинного обучения по-прежнему остается лучшим вариантом для многих сложных проектов.

Что вы узнали

В этой главе вы узнали, что знакомая всем с детства игра «Крестики-нолики» на самом деле вот уже много десятилетий помогает людям изучать и понимать принципы машинного обучения. Вы обучили собственную модель машинного обучения распознавать числа и создали игровое поле, состоящее из пронумерованных ячеек, благодаря которым можно отслеживать ходы. Этот проект был основан на исследовании британского ученого Дональда Мичи, который создал свой проект MENACE с использованием спичечных коробков и стеклянных бусин. Каждый спичечный коробок представлял возможное состояние игры (похоже на то, как вы наполняли коллекции обучающих примеров). Количество бусинок в спичечных коробках равнялось тому, сколько раз этот пример встречался в одной из коллекций.

Вы также в очередной раз убедились, насколько важно собрать большое количество обучающих примеров, чтобы со временем все больше и больше улучшать модель машинного обучения, ведь после каждой игры коллекции наполняются ходами победителя, а, значит, модель постепенно становится лучше и ее становится все труднее обыграть. Вы также узнали о возможностях краудсорсинга и о том, что если поделиться своими разработками с большой группой людей, то это может существенно сэкономить время и силы.

В следующей главе вы узнаете, что может пойти не так в проектах машинного обучения.



ГЛАВА 14

Запутать компьютер

К этому моменту вы узнали, сколько всего замечательного и полезного можно сделать с помощью машинного обучения и как машинное обучение используется в реальных больших проектах. Но тем не менее, как вы уже могли убедиться, системы машинного обучения не идеальны и не всезнающи. Их поведение и качество зависят от обучающих данных, которые мы им предоставляем.

От того, как мы обучаем системы машинного обучения, зависит качество их ответов, и эти ответы не всегда верные. В этой главе вы узнаете о самой распространенной проблеме, с которой сталкиваются создатели проектов искусственного интеллекта, а именно о предвзятости.

Проект в этой главе основан на старой известной истории, которую часто рассказывают студентам, которые изучают искусственный интеллект. Скорее всего, эта история — выдумка, но она иллюстрирует влияние предвзятости на обучающие данные для моделей машинного обучения.

Вот пример того, как эту историю часто рассказывают:

Однажды армия США решила использовать машинное обучение для распознавания вражеских танков, прячущихся за деревьями в лесу. Исследователи обучили модель машинного обучения на примерах фотографий леса без танков и фотографий того же леса, но с танками, прячущимися за деревьями.

Модель, казалось, хорошо справлялась с распознаванием изображений, которые ей давали исследователи. Но когда армия США протестировала эту систему машинного обучения, полученные результаты были ненамного лучше случайных догадок.

Выяснилось, что в обучающих примерах фотографии замаскированных танков были сделаны в пасмурный день, а фотографии пустого леса — в солнечный день. То есть модель

машинного обучения научилась отличать пасмурные дни от солнечных, а распознавать замаскированные танки даже не собиралась.

А вот еще один пример того, как рассказывают эту историю:

Однажды армия США решила научить компьютер распознавать разницу между американскими и иностранными танками. Исследователи обучили модель машинного обучения на примерах официальных фотографий американских танков и шпионских фотографий иностранных танков.

Модель, казалось, хорошо справлялась с изображениями, которые ей давали исследователи. Но когда армия США протестировала эту систему машинного обучения, полученные результаты были ненамного лучше случайных догадок.

Оказалось, что в обучающих примерах исследователей фотографии американских танков были большими и очень детализированными, они имели высокое разрешение. Наоборот, шпионские фотографии иностранных танков, которые делались с большого расстояния, были маленькими, а изображения были размытыми.

Таким образом, модель машинного обучения научилась распознавать разницу между фотографиями плохого и хорошего качества, а не разницу между американскими и иностранными танками.

Еще один пример такой путаницы: когда исследователи из Стэнфордского университета разрабатывали систему машинного обучения для распознавания рака кожи по фотографиям, они случайно создали модель машинного обучения, которая распознает линейки. А все потому, что медицинские фотографии рака кожи обычно включают и линейку, показывающую размер поражения или опухоли.

Дело в том, что из-за своей непреднамеренной предвзятости системы машинного обучения могут научиться обнаруживать закономерности, о которых их создатели могли не знать или которые не должны были рассматриваться как закономерности.

В этой главе вы научите классификатор изображений распознавать изображения объектов. Но на этот раз вы добавите некоторую предвзятость, чтобы он ошибался. Так вы сможете сами увидеть, какие проблемы могут привести к ошибкам модели машинного обучения. А затем мы поговорим о том, как этих проблем можно избежать и как исправить и улучшить модель машинного обучения.



Выполнение проекта

Выберите два предмета, фотографии которых вы хотите научить компьютер распознавать. При этом выбирайте такие предметы, которые явно отличаются друг от друга. Также не выбирайте ничего слишком личного, так как для выполнения этого проекта вам нужно будет загрузить фотографии в Интернет.

Для своего примера я выбрал то, что нашел на своей кухне, — лимон и грейпфрут. Но вы можете выбрать буквально все что угодно.

Положите первый предмет на любую поверхность и сделайте 10 его фотографий. Вам не нужно, чтобы фотографии были в высоком разрешении. Лучше всего подойдут маленькие фотографии (менее 800 пикселей в ширину).

Я положил свой грейпфрут на деревянный пол в темной комнате и сделал фотографии, которые вы можете видеть на рисунке 14.1.

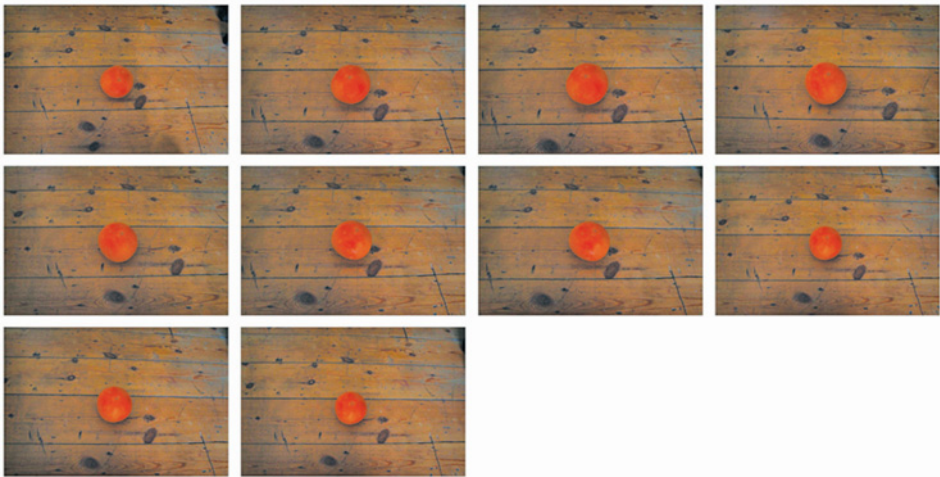


Рис. 14.1. Фотографии грейпфрута — первого предмета, который я выбрал для выполнения этого проекта

Теперь положите второй предмет в другое место и сделайте 10 его фотографий.

Лимон я положил на синий ковролин и сделал фотографии со вспышкой, которые вы можете видеть на рисунке 14.2.

Выбирайте такой ракурс, чтобы предметы на них не были слишком большими. Постарайтесь, чтобы фотографии каждого предмета были похожи между собой, примерно так, как показано на рисунках 14.1 и 14.2.

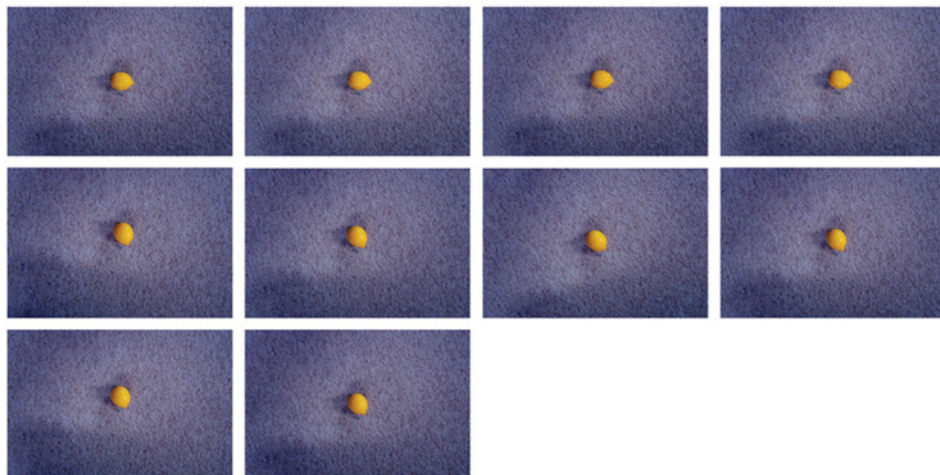


Рис. 14.2. Фотографии лимона — второго предмета, который я выбрал для выполнения этого проекта

На данном этапе ваша цель состоит в том, чтобы сделать все 10 фотографий для каждого из предметов очень похожими. Но при этом вам нужно сделать все, чтобы наборы фотографий двух предметов как можно больше отличались друг от друга — самим предметом, фоном, освещением и тому подобное. Например, если ваши фотографии первого предмета были сделаны на темном фоне, тогда сделайте фотографии второго предмета на светлом фоне. Но тогда все фотографии первого предмета должны быть на темном фоне, а все фотографии второго предмета — на светлом.

Вот несколько идей, которые вы можете использовать, чтобы сделать фотографии двух предметов наиболее различными.

Если все фотографии первого предмета...	... то все фотографии второго предмета нужно сделать...
на темном фоне	на светлом фоне
на плитке	на траве
очень яркие	в каком-то темном месте
четкие и сфокусированные	неясные и размытые
на улице, на фоне деревьев	в помещении, в интерьере одной из комнат

Взгляните еще раз на мои фотографии. Фотографии на рисунке 14.1 все выполнены при недостатке освещения, на фоне темно-коричневого деревянного пола. Фотографии на рисунке 14.2 сделаны со вспышкой, на поверхности однотонного ковролина.

Только, пожалуйста, не используйте идеи моих фото! Будьте креативными, придумайте что-то свое!

После того как вы сделаете свои 20 фотографий, вам нужно будет разместить их где-нибудь в Интернете. Выберите любой веб-сервис для размещения фотографий, который позволит вам бесплатно загружать фотографии в Интернет. Если даже у вас есть учетная запись на каком-либо хостинге, рекомендую сделать новую: вряд ли 20 одинаковых фотографий случайного предмета понравятся вашим подписчикам.

Самое главное — загрузить свои фотографии на такой ресурс, к которому можно было бы получить доступ без входа в систему. Так ваша система машинного обучения сможет всегда получать к ним доступ и учиться на них.

Обучение модели

1. Перейдите по ссылке <https://machinelearningforkids.co.uk/> Создайте новый проект машинного обучения. Задайте ему имя «Запутать компьютер», а в качестве распознаваемого типа данных выберите «изображения».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2.

2. Нажмите на кнопку «Обучить», как показано на рисунке 14.3.

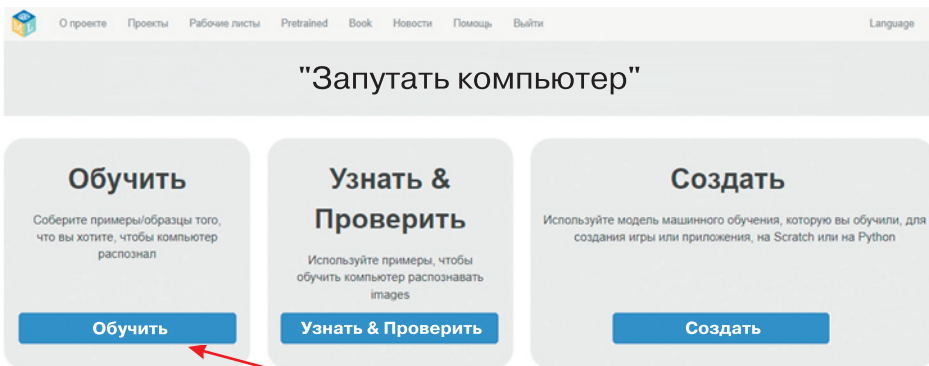


Рис. 14.3. Нажмите на кнопку «Обучить», чтобы подготовить коллекции обучающих примеров

3. Нажмите на кнопку «Добавить новую метку», чтобы создать две коллекции изображений, как показано на рисунке 14.4. Назовите их так же как и предметы, которые вы выбрали для распознавания. Имена коллекций не влияют на обучение модели, но полезны для вас, чтобы вы не запутались. Я назвал свои коллекции «grapefruit» (англ.: грейпфрут) и «lemon» (англ.: лимон).

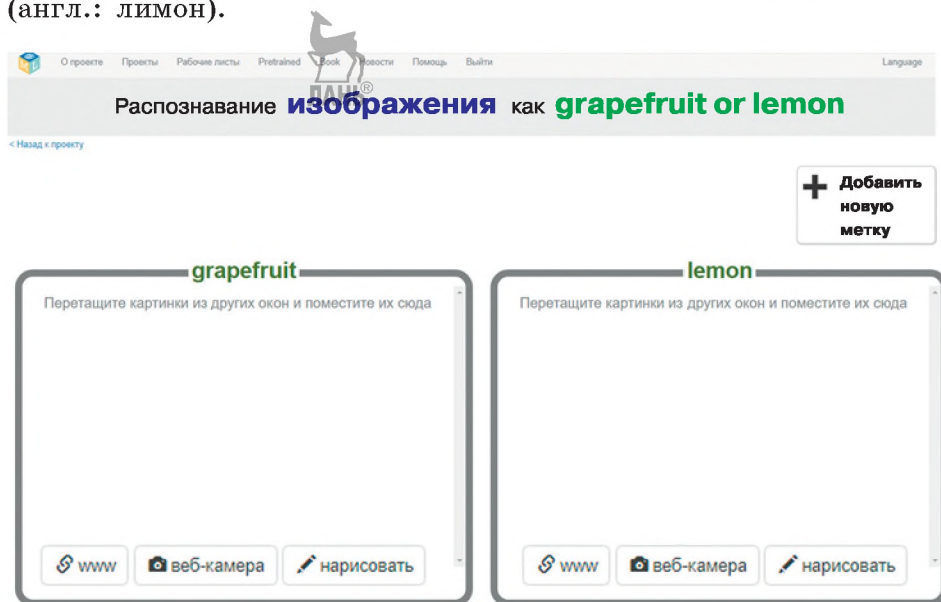


Рис. 14.4. Создайте две коллекции обучающих примеров для двух предметов, которые вы выбрали для распознавания

4. Добавьте обучающие примеры изображений в каждую из коллекций. Чтобы это было легче сделать, разместите рядом два окна браузера, как показано на рисунке 14.5. В одном окне должны быть ваши коллекции, а в другом — сайт, на которые вы ранее загрузили фотографии своих предметов.

Перетащите фотографии с этого сайта в ваши коллекции обучающих примеров.

5. Повторяйте шаг 4 до тех пор, пока все 20 фотографий не окажутся в нужных коллекциях (рис. 14.6).

6. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

7. Нажмите на кнопку «Узнать и проверить», чтобы перейти к следующему этапу проекта.

8. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 14.7).

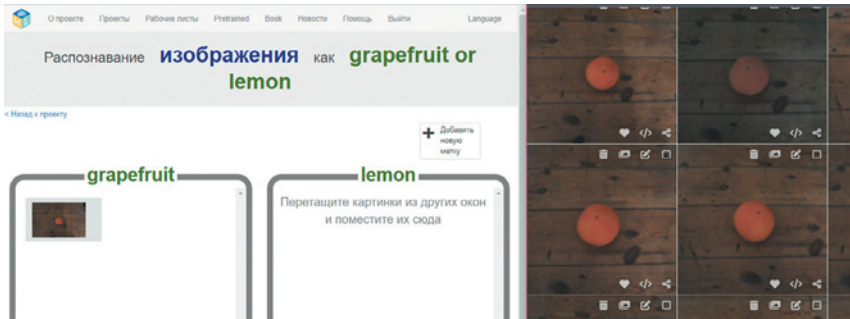


Рис. 14.5. Расположите два окна браузера рядом и перетащите фотографии в коллекции обучающих примеров

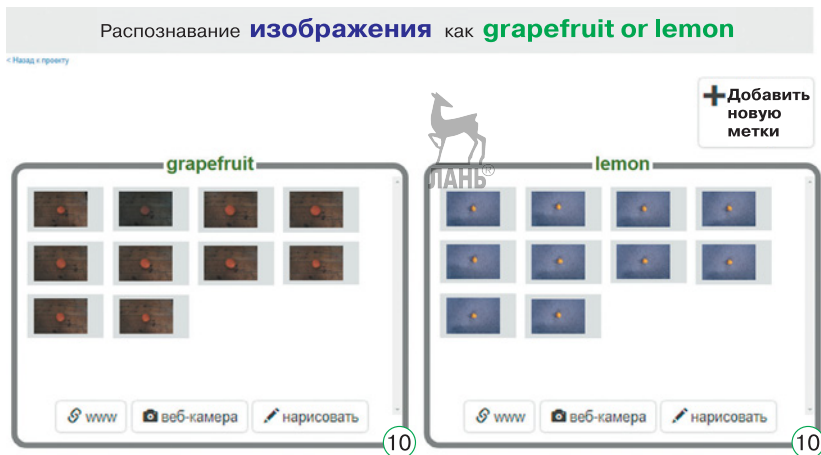


Рис. 14.6. Перетащите все ваши фотографии в коллекции обучающих примеров

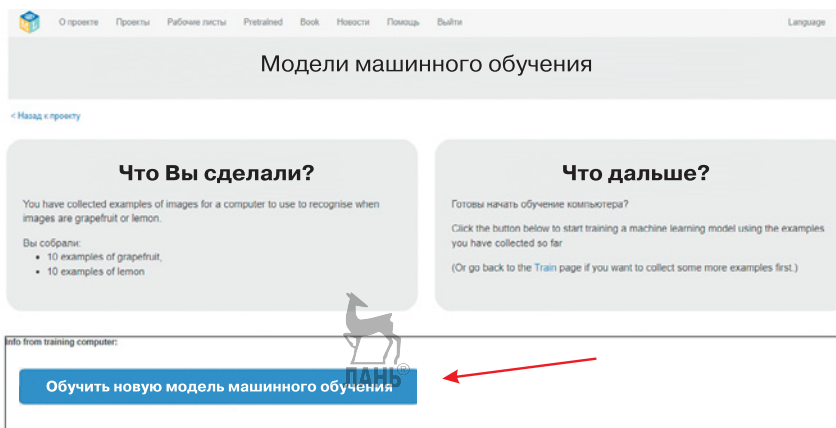


Рис. 14.7. Обучите свою модель машинного обучения

Вашей модели понадобится некоторое время, чтобы обучиться. Пока вы ждете, можете перейти к следующему шагу — подготовке проекта.

Подготовка проекта

Сделайте еще несколько фотографий каждого из предметов, но на этот раз поменяйте их фоны местами.

Иными словами, вам нужно сделать фотографии первого предмета там, где вы фотографировали второй. А фотографии второго предмета должны быть сделаны там, где в прошлый раз вы фотографировали первый.

В моем примере это означает, что фотографии лимона нужно сделать на плохо освещенном деревянном полу, а фотографии грейпфрута — на хорошо освещенном однотонном ковре. Сравните мои фотографии на рисунке 14.8 с фотографиями на рисунках 14.1 и 14.2.

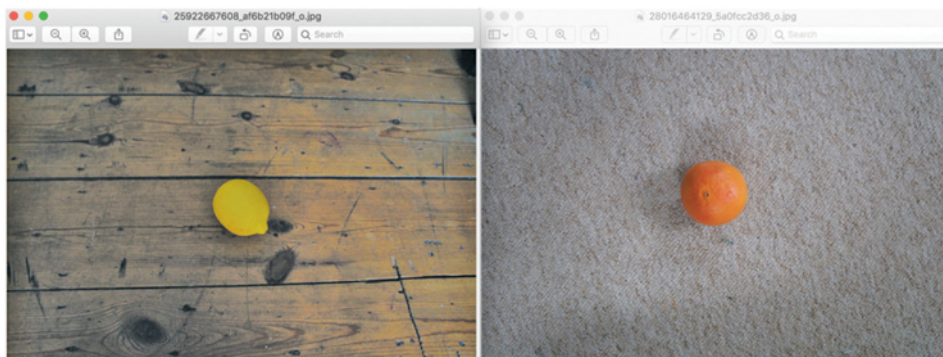


Рис. 14.8. Поменяйте местами предметы и сфотографируйте их снова

Эти фотографии вам не нужно никуда загружать. Вам только нужно сохранить их на свой компьютер, чтобы использовать во время тестирования.

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы, а затем нажмите на кнопку «Создать» (рис. 14.9).

2. Нажмите на кнопку «Scratch 3» (рис. 14.10).

3. Нажмите на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch.

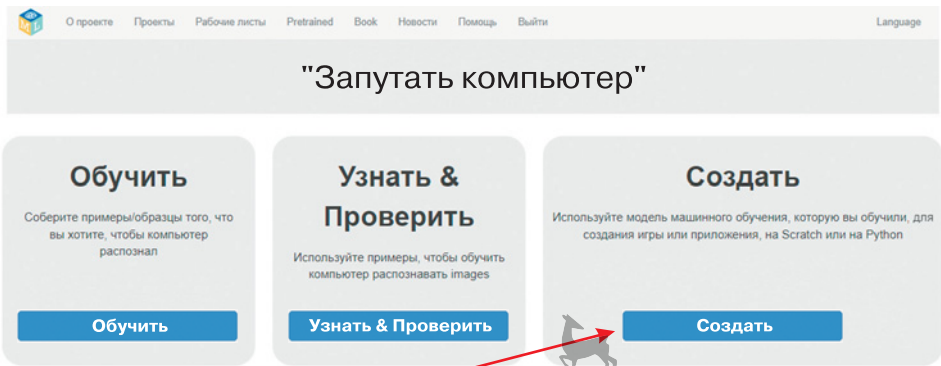


Рис. 14.9. Настало время протестировать ваш проект!

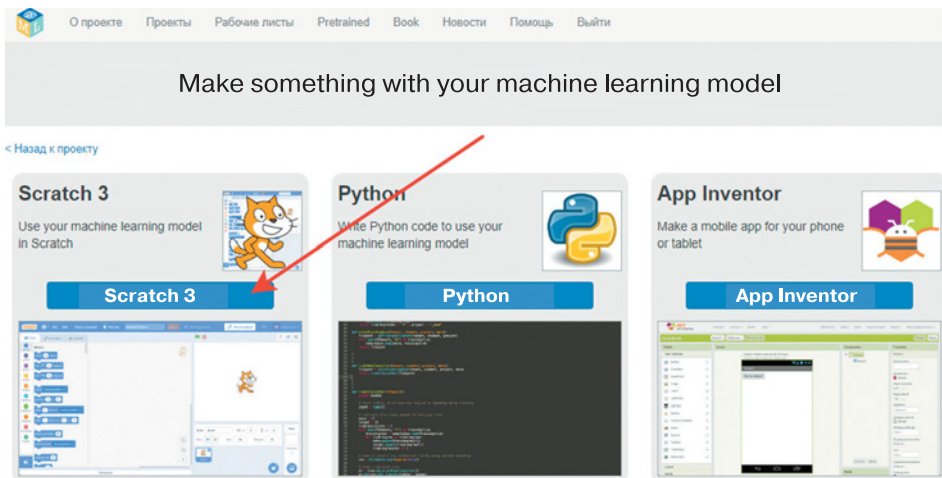


Рис. 14.10. Нажмите на кнопку «Scratch 3», чтобы протестировать вашу модель машинного обучения

4. Наведите указатель мыши на иконку «Выбрать спрайт» в правом нижнем углу экрана (иконка с мордочкой кота). Затем нажмите на кнопку «Загрузить спрайт» (рис. 14.11). Загрузите одну из двух ваших новых фотографий.

5. Создайте такой же скрипт, как показан на рисунке 14.12. Этот скрипт пытается распознать изображение костюма спрайта и выводит на экран решение модели, **т.е.** что она распознала на этой фотографии.

6. Снова нажмите на иконку «Загрузить спрайт» и загрузите вашу вторую тестовую фотографию. Создайте такой же скрипт, как и в предыдущем шаге. Он показан на рисунке 14.13.

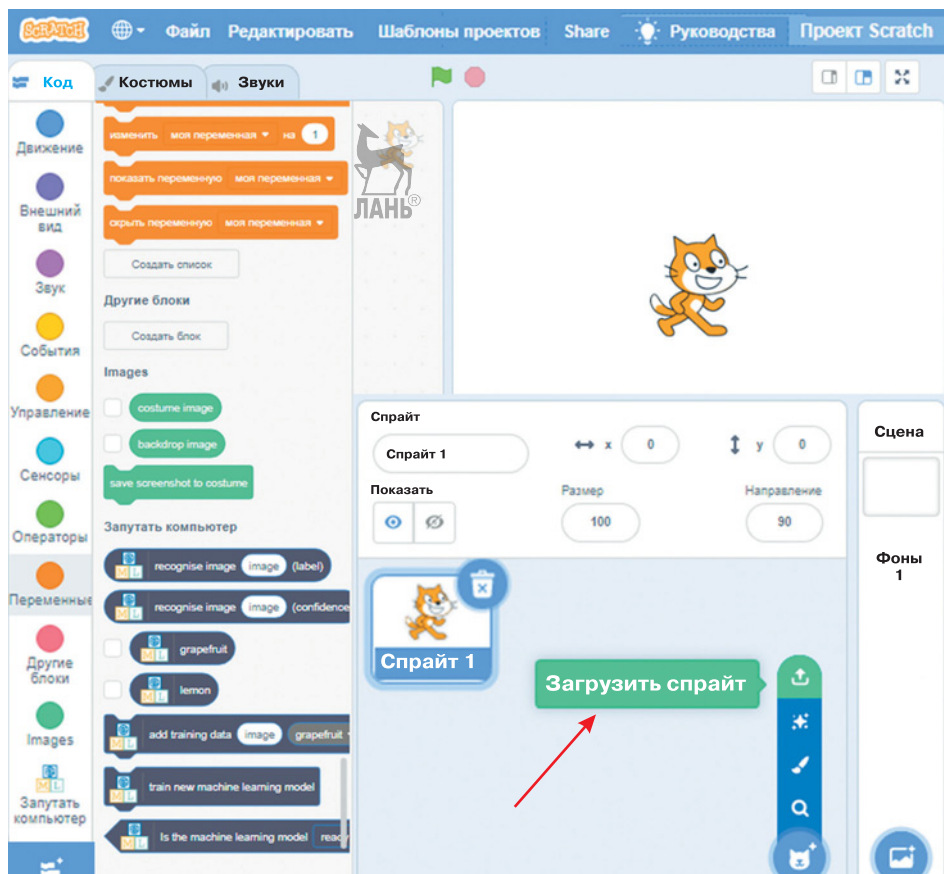


Рис. 14.11. Загрузите новый спрайт

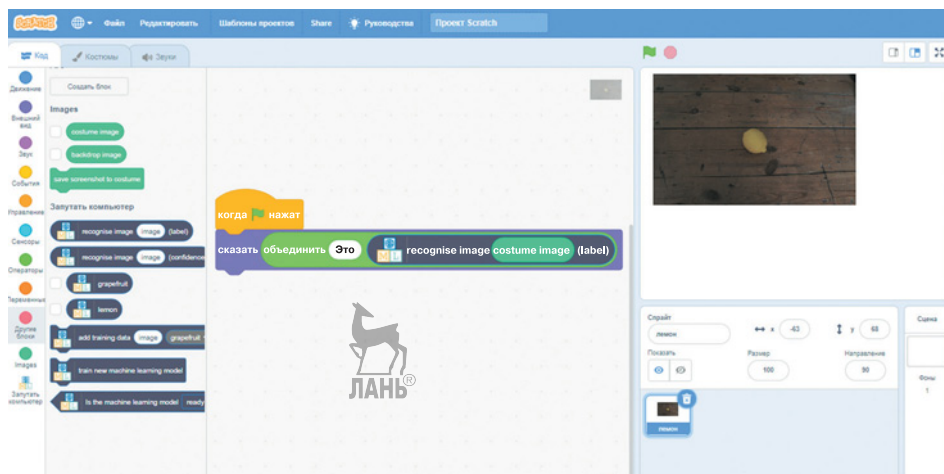


Рис. 14.12. Создайте небольшой скрипт для тестирования модели

Как и ожидалось, ваша модель, скорее всего, дала неверный ответ.

Казалось бы, я обучил модель машинного обучения с использованием современных передовых технологий, но она не смогла отличить лимон от грейпфрута. Хотя любой человек сделал бы это с легкостью.

Как вы считаете, что пошло не так?

Обзор и исправление проекта

Есть сразу несколько причин, по которым моя модель дала неверный ответ.

Во-первых, подумайте о композиции моих фотографий. На всех фотографиях предмет занимает около 5% площади всей фотографии. То есть около 95% площади каждой фотографии занимает фон. Это схематично показано на рисунке 14.15.

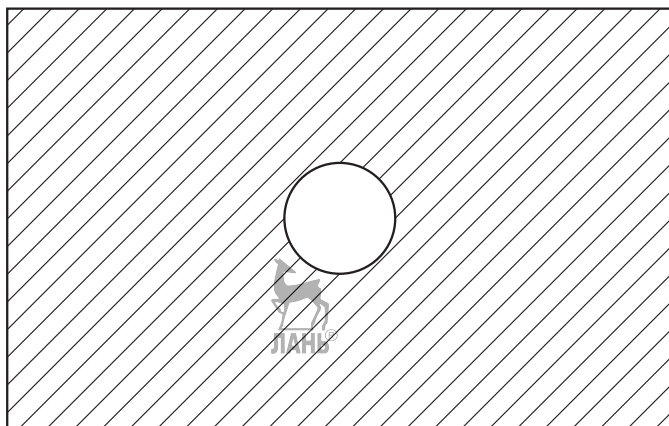


Рис. 14.15. Оказывается, большую часть моих фотографий занимает фон, а не предмет

Когда вы собирали обучающие примеры для своей модели машинного обучения, по сути, вы хотели, чтобы компьютер смог распознать нечто общее у всех этих фотографий. А затем компьютер должен будет распознавать эти же предметы и на других фотографиях, когда увидит что-то похожее по характеристикам.

Но что же получилось? Когда я тестировал модель на примере фотографии лимона, около 95% площади этой фотографии были

очень похожи на 95% площади всех обучающих примеров для грейпфрута (см. рис. 14.1). А когда я обучал свою модель машинного обучения, я никак не дал ей понять, что интересующая меня часть фотографии — это всего лишь 5% в центре, а не все остальное пространство.

Если вы расположите рядом два окна браузера (в одном — обучающие примеры для грейпфрута, а в другом — тестовый пример лимона), вы сразу же поймете, почему модель дала именно такой ответ (рис. 14.16).



Рис. 14.16. Сравните обучающие примеры с тестовой фотографией, и вы сразу же поймете, почему модель ошиблась

То есть, если рассматривать фотографии в целом, бóльшая часть моей тестовой фотографии (справа на рисунке 14.16) очень похожа на бóльшую часть каждого обучающего примера, которые я обозначил как «грейпфрут».

Системы машинного обучения созданы так, чтобы учиться распознавать общие закономерности в тех обучающих примерах, которые вы им даете. Но иногда получается так, что выявленные закономерности, о которых задумали вы (или которые вы бы распознали сами), оказываются совсем другими.

Как вы думаете, как можно исправить этот проект?

Есть несколько решений, которые вы можете использовать в вашем проекте. Например, добавить новые обучающие примеры — фотографии, на которых ваш предмет составляет бóльшую часть всего изображения, как показано на рисунке 14.17.

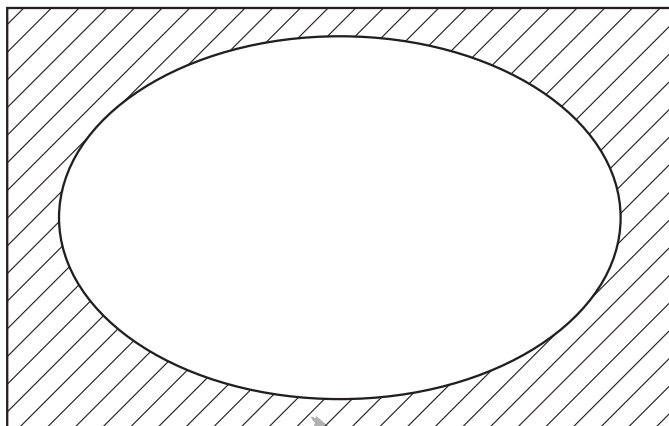


Рис. 14.17. Попробуйте добавить фотографии, на которых ваш предмет выглядит большим (занимает большую часть фотографии)

Но это может сработать только для тех проектов, в которых вы можете добиться того, чтобы на всех тестовых изображениях предмет был бы одинакового размера.



Примечание

Единственное, что должно быть общим в ваших обучающих примерах, — это те атрибуты, которые вы хотите, чтобы модель машинного обучения научилась распознавать.

Для данного проекта лучший способ убедиться в этом — сделать много фотографий двух предметов в разных местах, с разными фонами, освещением, размерами, углами обзора и ориентацией фотографий. Измените все, что вы только можете придумать, чтобы единственное, что оставалось бы общим у всех фотографий, был сам предмет.

Например, на рисунке 14.18 показаны хорошие варианты обучающих примеров для распознавания грейпфрута.

Изменение фона, освещения и степени фокусировки для фотографий обучающих примеров — хорошее начало обучения модели для определения только самого грейпфрута как нечего общего на всех изображениях.

А можно сделать обучение еще лучше. Например, сейчас на всех обучающих примерах грейпфрут находится в одном и том же положении, и он примерно одинакового размера. Это нормально, если я могу гарантировать, что предметы на моих тестовых фотографиях будут иметь те же размер и положение. Но

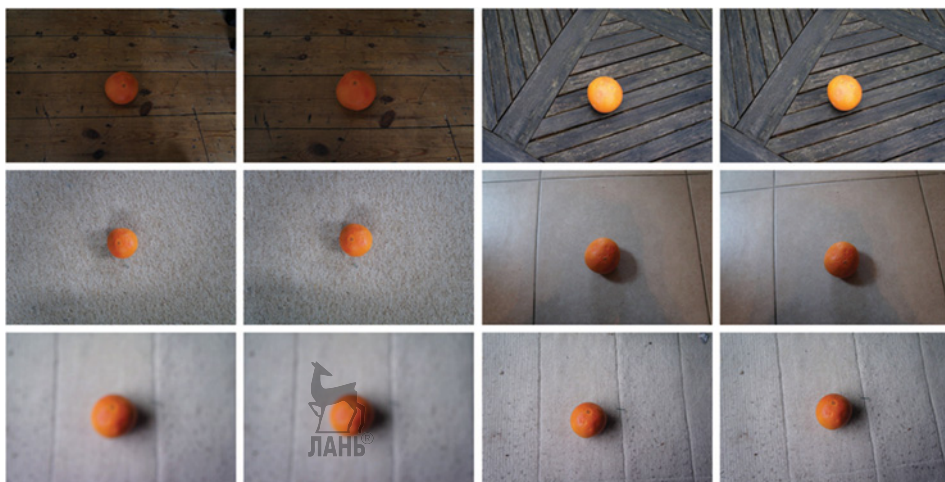


Рис. 14.18. Более удачные обучающие примеры для моего проекта

для создания действительно умной модели мне следует также добавить фотографии более и менее крупных, снятых в разных ракурсах грейпфрутов.

Попробуйте улучшить ваши обучающие примеры, а затем обучить модель заново.

Прошла ли ваша модель испытание после того, как вы разнообразили обучающие примеры?

Что вы узнали



В этой главе вы узнали, насколько важное значение имеет разнообразие в подходах к обучению моделей машинного обучения для предупреждения неожиданных ошибок. Будь то военный проект, который случайно распознает погоду вместо замаскированных танков, университетский исследовательский проект, изобретающий распознаватель линейки вместо классификатора рака кожи, или просто система, которая не может отличить грейпфрут от лимона. Теперь вы знаете о влиянии *непреднамеренной предвзятости*, которая может встречаться в наборах данных, используемых для обучения модели машинного обучения.

В следующей главе вы узнаете, что такое намеренная предвзятость и как она влияет на проекты машинного обучения.



ГЛАВА 15

Этические вопросы использования искусственного интеллекта

В предыдущей главе вы увидели, как случайно можно обучить систему машинного обучения таким образом, что она будет давать неверные ответы. Для этого нужно лишь внести предвзятость в ваши обучающие примеры.

В этой главе мы рассмотрим, как предвзятость иногда добавляется преднамеренно, чтобы влиять на ответы, которые дает система машинного обучения. Вы создадите приложение, которое будет рекомендовать людям фильмы на основе того, какие фильмы им нравятся. Но вы будете обучать свою модель таким образом, чтобы преднамеренно повлиять на ее рекомендации.

Выполнение проекта

Прежде всего в этом проекте нужно будет создать библиотеку фильмов, из которой ваше приложение будет выбирать свои рекомендации для пользователей. Для начала выберите три фильма, чтобы добавить их в библиотеку.



Рис. 15.1. Фильмы, которые я выбрал для своего проекта

В моем примере я захотел создать приложение, которое помогало бы людям рекомендовать старые классические фильмы. Поэтому я выбрал три фильма 1920-х годов, как показано на рисунке 15.1. Но вам совсем не обязательно делать то же самое, вы можете выбрать для своего проекта и современные фильмы.

Постарайтесь выбрать три совершенно разных фильма, которые могут понравиться самым разным людям.

Я выбрал фантастический фильм «Метрополис», комедию «Золотая лихорадка» и фильм ужасов «Носферату».

Обучение модели



1. Перейдите по ссылке <https://machinelearningforkids.co.uk/> Создайте новый проект машинного обучения и назовите его «Предвзятость». В качестве распознаваемого типа данных выберите «текст».



Примечание

Если вы не помните, как создать проект, вернитесь к разделу «Создание нового проекта» в главе 2. Желательно работать в гостевом режиме, не входя в аккаунт.

2. Нажмите на кнопку «Обучить», как показано на рисунке 15.2.

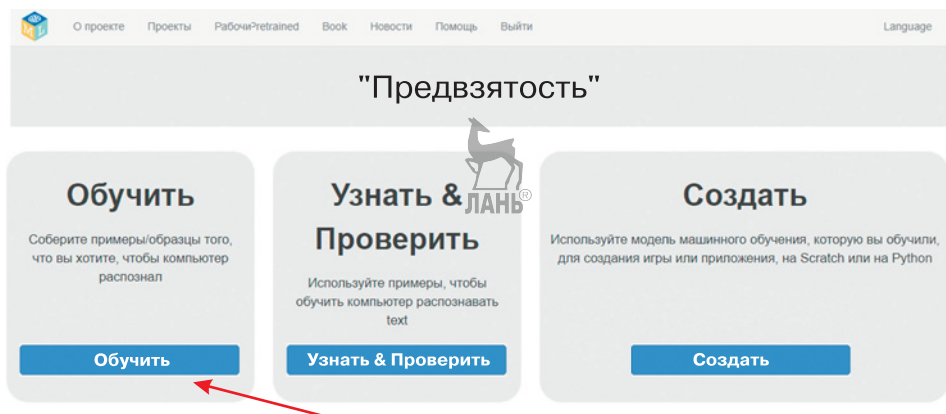


Рис. 15.2. Переход к первому этапу проекта машинного обучения — этапу «Обучить»

3. Нажмите на кнопку «Добавить новую метку» (рис. 15.3) и создайте три коллекции — по одной для каждого из выбранных фильмов.

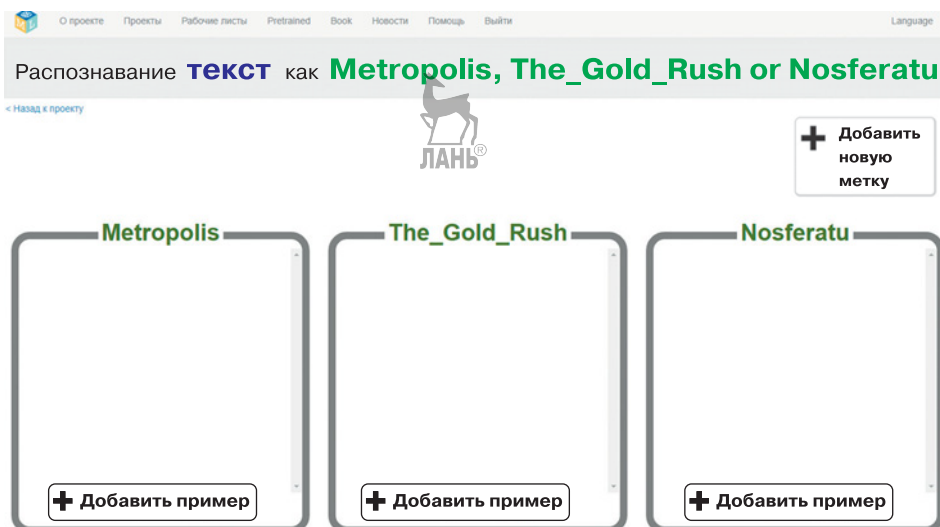


Рис. 15.3. Создайте коллекцию обучающих примеров для каждого из выбранных фильмов

4. Нажмите на кнопку «Добавить пример», чтобы добавить новый обучающий пример в коллекцию первого фильма (рис. 15.4). Напишите что-нибудь, что, по вашему мнению, мог бы сказать человек, которому понравился ваш первый фильм.

Например, мой первый фильм «Метрополис» — это научно-фантастический фильм, действие которого происходит в будущем. Поэтому я написал в качестве обучающего примера фразу «я люблю футуристические фильмы».

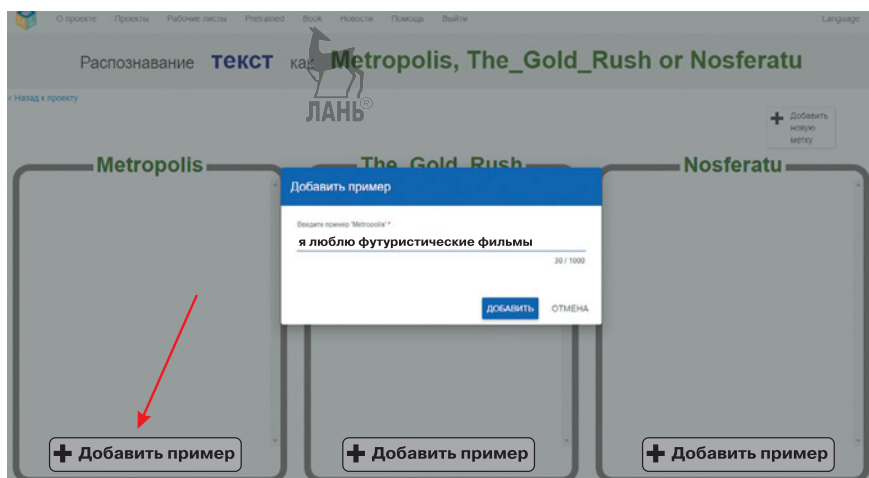


Рис. 15.4. Добавьте новый обучающий пример — фразу, которую мог бы сказать человек, которому понравился ваш первый фильм

5. Нажмите на кнопку «Добавить».

6. Повторяйте шаги 4 и 5 до тех пор, пока в каждой из трех коллекций не наберется хотя бы по пять обучающих примеров, как показано на рисунке 15.5.



Рис. 15.5. Добавьте по пять обучающих примеров в коллекцию каждого фильма

7. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.

8. Нажмите на кнопку «Узнать и проверить», чтобы перейти к следующему этапу проекта.

9. Нажмите на кнопку «Обучить новую модель машинного обучения» (рис. 15.6).

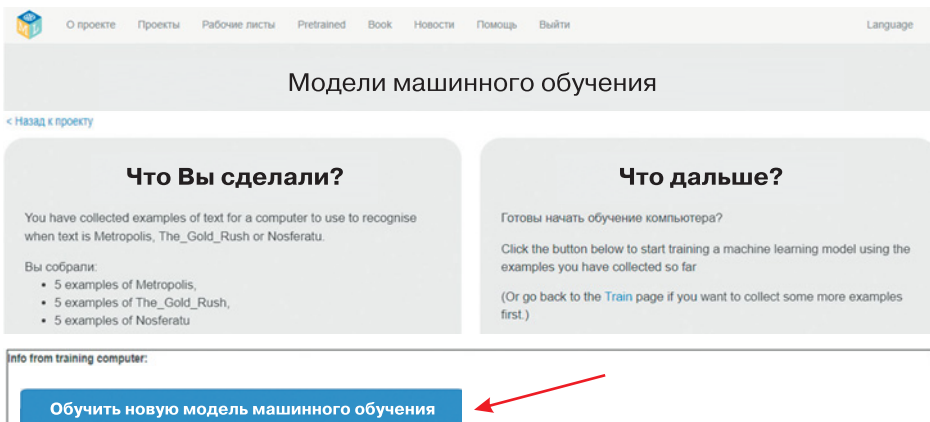


Рис. 15.6. Нажмите на кнопку «Обучить новую модель машинного обучения», чтобы создать новую модель и начать обучение

Компьютеру может понадобиться несколько минут, чтобы обработать ваши обучающие примеры, создать новую модель машинного обучения и обучить ее на этих примерах. Вы можете пока перейти к следующему шагу работы над проектом.

Подготовка проекта

Ура! Теперь у вас есть обученная модель машинного обучения. Настало время создать приложение для рекомендации фильмов, которое будет использовать эту модель.

1. Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы.
2. Нажмите на кнопку «Создать».
3. Нажмите на кнопку «Scratch 3», а затем на кнопку «Открыть в Scratch 3», чтобы открылось новое окно со средой Scratch.
4. Перейдите на вкладку «Костюмы» в левом верхнем углу. Затем наведите указатель мыши на иконку «Выбрать костюм» в левом нижнем углу экрана. Нажмите на кнопку «Загрузить костюм», чтобы загрузить постер вашего фильма, как показано на рисунке 15.7.

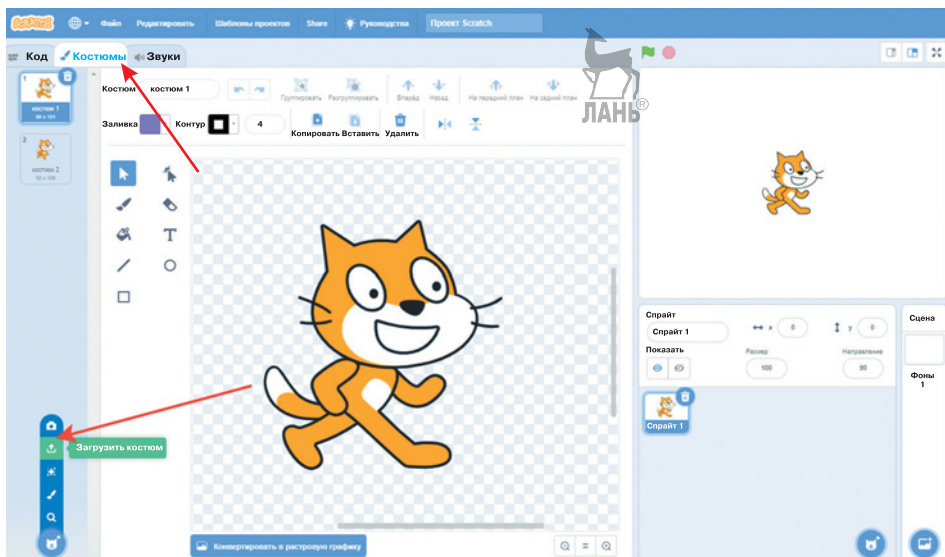


Рис. 15.7. Загрузите костюм



Примечание

Чтобы вспомнить, как сохранять изображения из Интернета на свой компьютер, вернитесь к шагу 1 главы 5 (с. 70). Если же вы не сохранили изображения постера фильма на своем компьютере, вы можете нажать на кнопку «Рисовать» в меню «Выбрать костюм». А затем нарисовать свой постер или просто написать название фильма.

5. Загрузите изображение постера первого из ваших фильмов, как показано на рисунке 15.8.

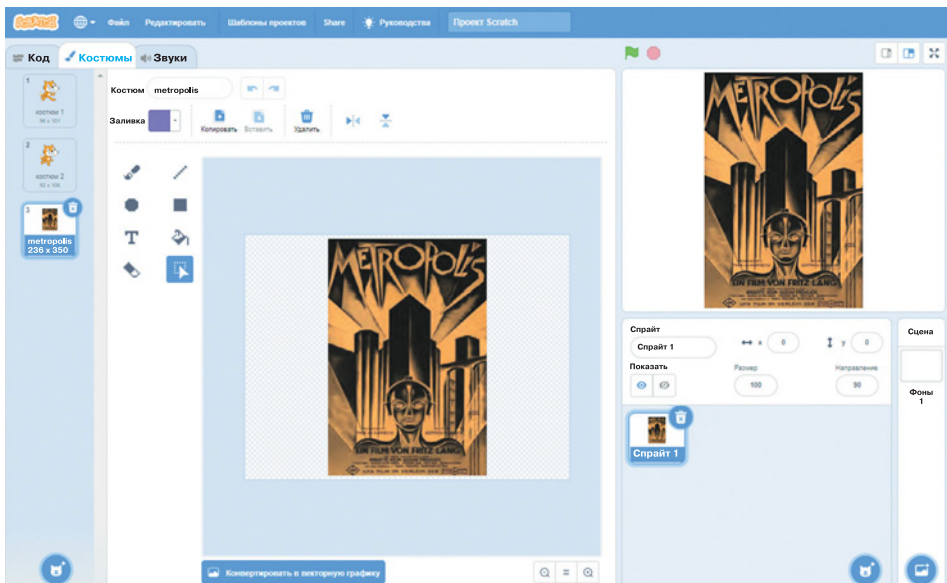


Рис. 15.8. Создайте костюм, который будет представлять ваш первый фильм

6. Повторяйте шаги 4 и 5, чтобы добавить в качестве костюмов постеры остальных двух фильмов. Будьте очень внимательны — вам нужно добавлять костюмы к одному спрайту, а не создавать новые спрайты! В итоге это должно выглядеть как на рисунке 15.9.

Назовите каждый костюм так же как и фильм, который он представляет.



Примечание

Еще раз убедитесь, что вы добавляете постеры как костюмы к одному спрайту, а не создаете новые спрайты. Это очень важно! Если вы сделали все правильно, ваш проект должен выглядеть так, как показано на рисунке 15.9.

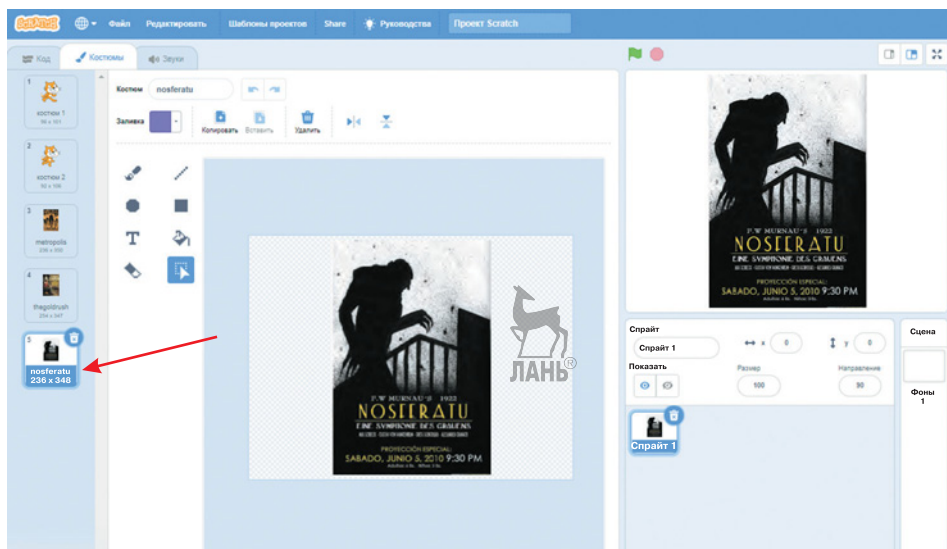


Рис. 15.9. Создайте костюм для каждого из фильмов

7. Перейдите на вкладку «Код» и создайте такой же скрипт, как показан на рисунке 15.10. Обратите внимание: вам нужно будет изменить названия фильмов на свои.

Этот скрипт будет опрашивать пользователя, какие фильмы ему нравятся, а затем использовать вашу модель машинного обучения, чтобы порекомендовать три фильма из вашей библиотеки фильмов.

8. Теперь займитесь дизайном приложения. Сделайте так, чтобы ваш проект выглядел так, как, по вашему мнению, должно выглядеть приложение, которое рекомендует пользователям фильмы. Обновите фон и спрайт. Вы можете использовать редактор рисования (как с ним работать, описано в главе 3 на с. 48), делать фотографии с помощью веб-камеры (глава 4, с. 58), загружать изображения, которые вы сохранили на компьютер (глава 5, с. 76), или выбрать готовый дизайн из библиотеки Scratch (глава 5, с. 79). Будьте креативными, проявите фантазию! На рисунке 15.11 показано, что получилось у меня.

9. Сохраните получившийся проект. Для этого в Главном меню выберите **Файл → Сохранить на свой компьютер**.



Тестирование проекта

Нажмите на зеленый флажок и протестируйте свой проект.

Попробуйте ввести различные фразы, описывающие сюжеты фильмов, которые вам нравятся, и посмотрите, что порекомен-

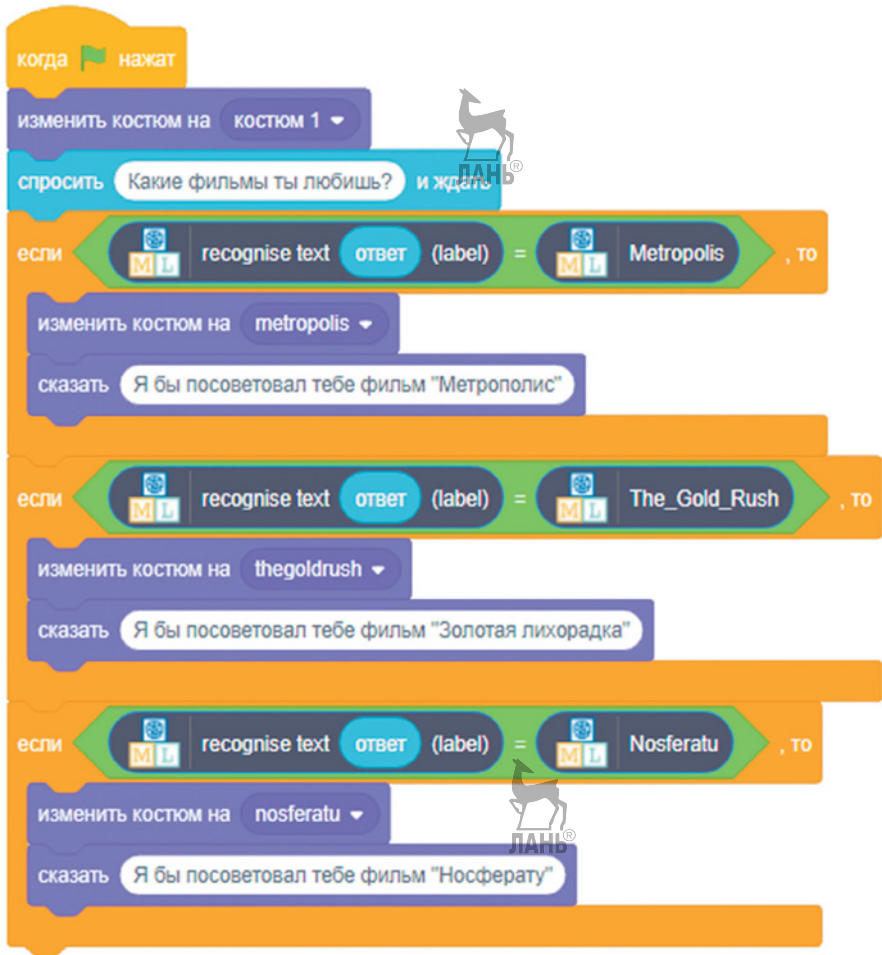


Рис. 15.10. Создайте такой же скрипт, не забудьте изменить названия фильмов

дует ваш проект. Избегайте использования слов или фраз, которые вы сохранили в качестве обучающих примеров. Потому что вам нужно проверить, научилась ли ваша модель машинного обучения распознавать именно новые фразы, а не уже знакомые.

Добавление предвзятости

Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы, а затем на кнопку «Обучить», чтобы вернуться к этапу обучения.

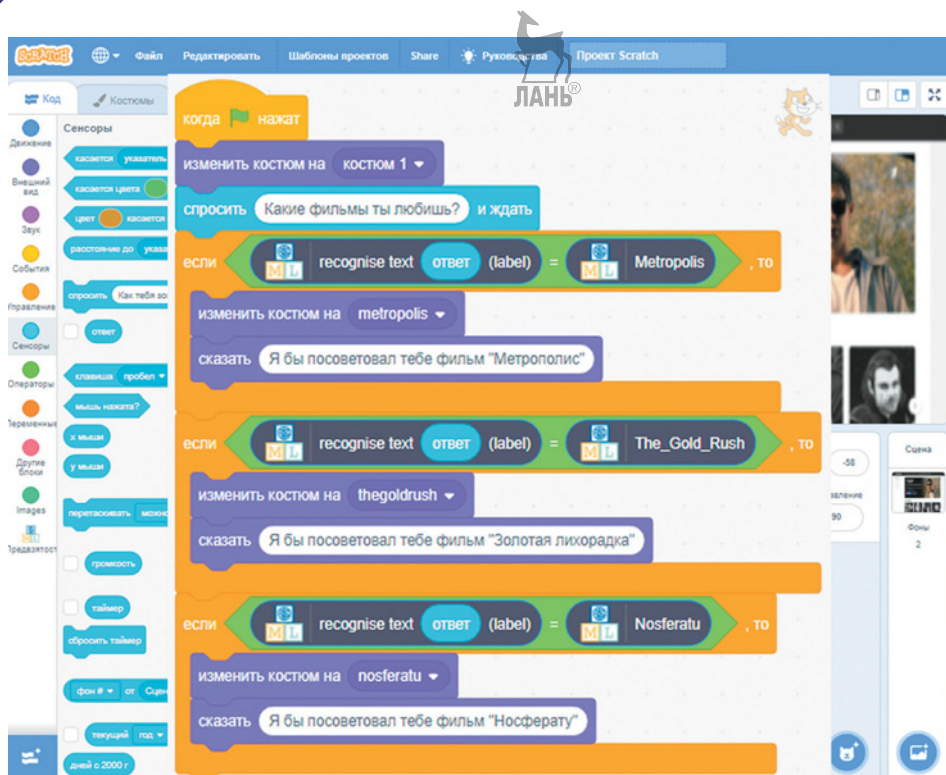


Рис. 15.11. Займитесь дизайном вашего приложения для рекомендации фильмов

Теперь выберите еще один (четвертый) фильм, который чем-то похож на один из трех уже имеющихся в вашей библиотеке. Для своего проекта я выбрал фильм «Франкенштейн» — это фильм ужасов, который немного похож на фильм «Носферату», который уже есть в моей библиотеке.

Нажмите на кнопку «Добавить новую метку», чтобы добавить еще одну коллекцию обучающих примеров — для четвертого фильма, который вы выбрали.

Удалите несколько обучающих примеров из коллекции того фильма, который похож на новый (в моем примере это «Носферату»), и добавьте эти же примеры в коллекцию четвертого фильма (в моем примере это «Франкенштейн»).

У вас должно получиться что-то похожее на рисунок 15.12.

Добавьте еще 12 обучающих примеров в коллекцию четвертого фильма. В итоге у вас должно получиться что-то похожее на рисунок 15.13.

Распознавание **текст** как **Metropolis, The_Gold_Rush or 2 other classes**

Назад к проекту

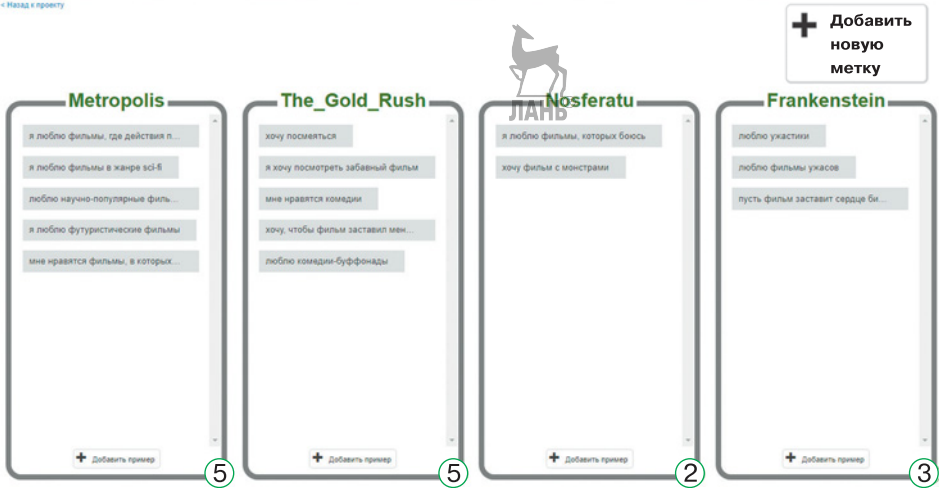


Рис. 15.12. Переместите несколько обучающих примеров из коллекции одного из старых фильмов в коллекцию нового фильма

Распознавание **текст** как **Metropolis, The_Gold_Rush or 2 other classes**

Назад к проекту

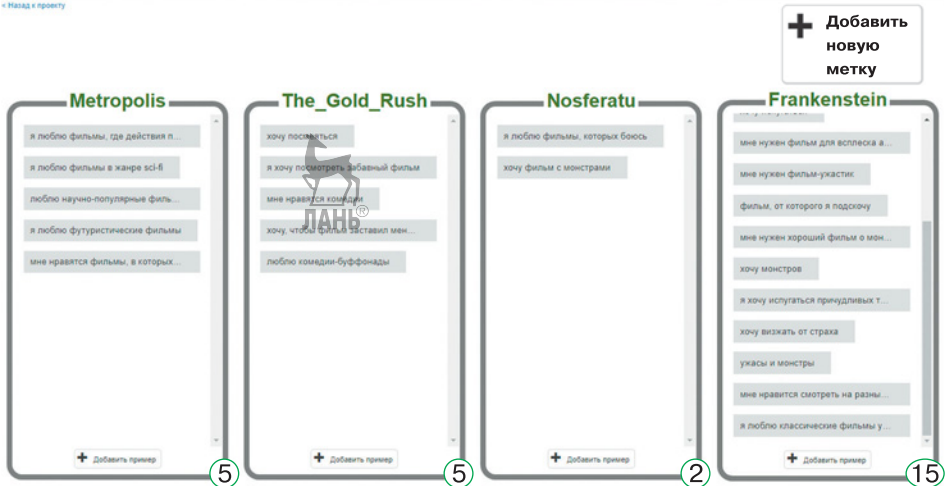


Рис. 15.13. Обучающие примеры для коллекции нового (четвертого) фильма

Нажмите на ссылку «Назад к проекту» в левом верхнем углу страницы. Затем нажмите на кнопку «Узнать и проверить». Обучите свою модель снова, в этот раз — на обновленном наборе обучающих примеров. Когда обучение модели закончится,

снова нажмите на ссылку «Назад к проекту», затем на кнопку «Создать», а потом снова на кнопку «Scratch 3».



Примечание

Прежде чем это сделать, не забудьте закрыть то окно Scratch, которое было открыто до этого. Только после этого сможет открыться проект, в котором появятся блоки вашего проекта машинного обучения.

В открывшемся окне Scratch выполните **Файл → Загрузить с компьютера** и выберите файл проекта, который вы скачали ранее. Обновите его так, чтобы в скрипте появился новый фильм, который вы только что добавили. И не забудьте добавить новый костюм для спрайта. Изменения в скрипте показаны на рисунке 15.14.

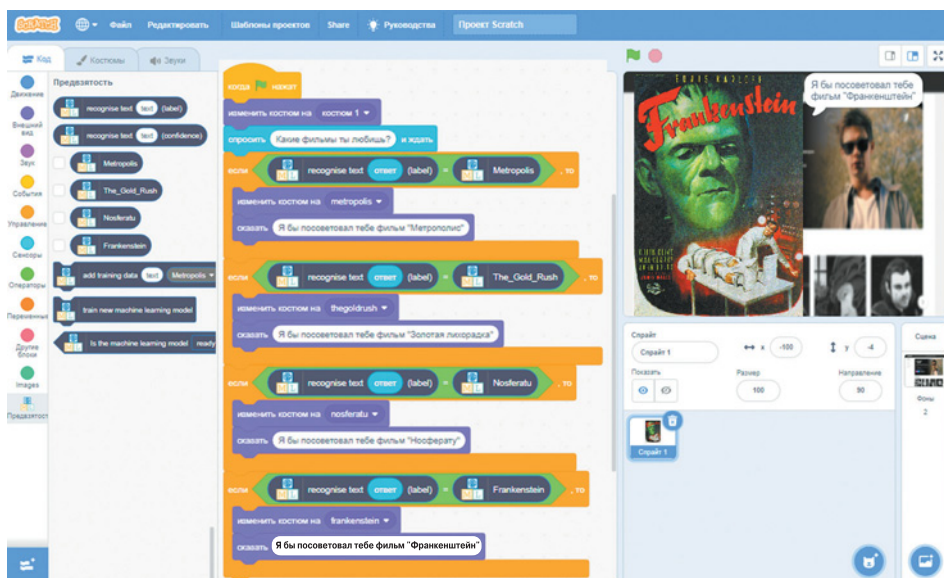


Рис. 15.14. Обновите проект Scratch так, чтобы добавить в него новый фильм

Тестирование проекта с добавленной предвзятостью

Попробуйте снова запустить и протестировать ваш проект. Вы наверняка обнаружите, что он отдает предпочтение новому (четвертому) фильму гораздо чаще, чем тому фильму, который похож на него.

В моем примере это проявлялось так. Если я в тестовом примере упоминал что-нибудь о фильмах ужасов, учащем сердцебиении, выбросе адреналина или монстрах, моя модель машинного обучения всегда рекомендовала «Франкенштейна», а не «Носферату», как это было раньше. При этом невозможно было найти никакого, казалось бы, логического объяснения, почему программа периодически рекомендует один и тот же фильм. Мне показалось, что модель предпочитает (или предвзято относится) к «Франкенштейну». А получилось так, что ее вообще было трудно заставить рекомендовать «Носферату», если только не набрав в точности одну из фраз, которую я использовал в качестве обучающего примера для «Носферату».

Поэкспериментируйте с тестовыми примерами и проверьте, как работает ваша модель машинного обучения. На самом деле каждая модель машинного обучения ведет себя немного по-своему, поэтому постарайтесь догадаться, чем руководствуется именно та модель, которую обучили лично вы.

Обзор проекта

В главе 8 мы говорили о показателях эффективности модели машинного обучения, таких как точность и полнота. Если вы рассчитаете некоторые из этих значений до и после того, как преднамеренно добавите предвзятость в модель машинного обучения, то вы сможете количественно оценить, как предвзятость влияет на ваш проект.

Когда вы решите, что поняли, как ведет себя и чем руководствуется ваша модель машинного обучения, следующим шагом будет понять почему. Посмотрите еще раз на рисунок 15.13. Почему моя модель машинного обучения чаще рекомендует «Франкенштейна»?

Когда вы собираете обучающие примеры, вы просите компьютер найти закономерности в этих примерах, которые он будет использовать впоследствии, чтобы научиться распознавать тестовые примеры. Количество примеров, которые вы помещаете в каждую коллекцию, также существенный фактор ее обучения. Я повлиял на модель машинного обучения, когда загрузил больше примеров «Франкенштейна», таким образом, что она перескочила за границы обособленных областей примеров.

Представьте, что вы учите ребенка рекомендовать фильмы. А теперь представьте, что вы специально 5 раз говорите ему, что он должен порекомендовать фильм А, и 1000 раз говорите, что

он должен порекомендовать фильм Б. Как это повлияет на его поведение? Если вы снова и снова скажете ему, что правильный ответ — «Фильм Б», он, вероятно, запомнит, что фильм, который он должен рекомендовать, — это «Фильм Б».

Это похоже на то, как ведут себя системы машинного обучения. Компьютер ищет закономерности во всех ваших обучающих примерах самыми разными способами. Ваше обучение подсказывает ему, какому из этих способов следует доверять больше, чем другим. Если обучающие примеры снова и снова говорят ему, что закономерности, методы и процессы, которые приводят к ответу «Фильм Б», верны, он учится доверять этим закономерностям, методам и процессам. Если обучающие примеры снова и снова говорят ему, что закономерности, методы и процессы, которые приводят к ответу «Фильм А», неверны, он учится не доверять им.

Даже если он определяет закономерности, методы и процессы, которые в будущем приведут к ответу «Фильм А», ваши обучающие примеры научили его не доверять им и вместо этого предпочитать те, которые предлагают фильм Б.

Кроме того, как вы уже убедились, важным фактором при создании систем машинного обучения является объем обучающих примеров в каждой коллекции. Дисбаланс количества обучающих примеров в разных коллекциях может привести к тому, что мы называем *предвзятой системой машинного обучения*.

Случаи, когда предвзятость полезна

До сих пор в большинстве проектов этой книги мы с вами старались наполнять разные коллекции каждого проекта примерно одинаковым количеством обучающих примеров. Это общепринятый принцип, который используется во многих проектах машинного обучения в попытке свести предвзятость к минимуму.

Однако, хотя предвзятость и является важным фактором, о котором следует помнить, ее наличие не всегда плохо.

Предположим, что вы обучаете компьютер распознавать разницу между тремя возможными исходами: X, Y и Z. Представьте, что X и Y очень распространены, один из них почти всегда является правильным ответом. Исход Z возможен, но очень и очень редко. Несмотря на то что Z почти никогда не происходит, вы хотите научить компьютер распознавать его, когда это нужно.

Сбалансированный набор обучающих примеров с одинаковым количеством примеров для X, Y и Z здесь не поможет. Наличие

большого количества обучающих примеров для X и Y и меньшего для Z может обучить модель машинного обучения тому, что X и Y более вероятны, и в данном случае это правильно. Исходы X и Y встречаются чаще, а Z — редко. Такая система по-прежнему будет предвзятой, но эта предвзятость отражает статистическую вероятность различных исходов, поэтому она может оказаться уместной и полезной. Другими словами, предвзятость — это не всегда плохо.

Искусственный интеллект и этические вопросы

На протяжении всей этой главы вы видели, что способ обучения, который вы используете для своей модели машинного обучения, сильно влияет на ответы, которые она дает. Как вы думаете, как это влияет на обязанности людей, которые создают системы искусственного интеллекта? Считаете ли вы, что разработчики систем искусственного интеллекта несут этическую ответственность за сбалансированность обучающих данных или за создание предвзятых систем?

Имеют ли значение их намерения? Если кто-то случайно работает предвзятую систему, будет ли это более или менее этично? А если кто-то намеренно исказил свои обучающие данные, потому что хотел повлиять на результат работы своей системы?

Имеет ли значение в этом случае финансовая сторона вопроса? Другими словами, если, например, продюсер четвертого фильма из моего проекта заплатил мне много денег, чтобы мое приложение рекомендовало его фильм, а не фильмы конкурентов, было бы это более или менее этично, чем создание предвзятого приложения, которое не принесет личной выгоды мне как создателю?

Имеет ли значение тема разработки? Как вы считаете, предвзятое приложение (система искусственного интеллекта), рекомендуемое фильмы, вызывает меньше этических проблем, чем приложение, которое дает врачам рекомендации по лечению пациентов?

Представьте себе приложение (использующее модель машинного обучения), которое рекомендует, какие лекарства следует назначать пациентам. Каждая коллекция обучающих примеров — это тип лекарства, а содержащиеся в ней обучающие примеры — это медицинские записи пациентов, для которых это лекарство было лучшим лечением. Подобные системы широко используются сегодня. Системы машинного обучения могут

научиться распознавать закономерности в огромном количестве подробных медицинских записей и объединять их с доказательствами, полученными из такого же огромного количества медицинских исследований и литературы. Время таких помощников, конечно, еще не настало, но в ближайшие несколько лет использование подобных систем значительно возрастет.

Теперь, когда вы сами убедились, насколько легко повлиять на систему машинного обучения, чтобы она предпочла один ответ другому, не призадумались ли вы о том, а как должны использоваться такие системы? Теоретически же возможно, что некий производитель отдельно взятого лекарства может вознаградить разработчиков медицинского приложения за намеренную предвзятость их модели машинного обучения в отношении своего лекарства, а не лекарств других производителей.

Мы с каждым днем все больше полагаемся на системы машинного обучения для принятия важных решений, влияющих на жизнь людей. И не только в сфере медицины. Системы машинного обучения дают финансовые рекомендации, которые банки и кредитные компании используют, чтобы определить, следует ли предлагать кому-то страховку, могут ли они получить кредит или какую процентную ставку им следует назначить. Системы машинного обучения скоро будут управлять легковыми и грузовыми автомобилями на дорогах. И это далеко не все примеры.

Одним из способов защиты от таких этических проблем может быть принуждение компаний к прозрачности и раскрытию того, как обучаются их системы машинного обучения. Но вы сами видели, сколько усилий уходит на подготовку обучающих примеров. Компании тратят много времени и денег на подготовку обучающих данных, которые они собирают, чтобы сделать свои системы машинного обучения лучше, чем у их конкурентов. Поэтому вполне естественно их желание держать свои обучающие данные и способы их получения в секрете. А какое бы вы предложили решение, чтобы балансировать вопросы этики с коммерческими интересами компаний?

Как вы думаете, нужны ли какие-то законы и средства регулирования в отношении способов обучения или применения систем искусственного интеллекта? Если да, то должна ли политика этики использования искусственного интеллекта разрабатываться отдельными компаниями или же органами власти?

Эта глава заканчивается большим количеством вопросов. И, поверьте, таких вопросов гораздо больше, чем ответов на них, что отражает текущее состояние проблемы этики в использовании искусственного интеллекта. Системы машинного обучения

могут улучшить нашу жизнь, обучая компьютеры делать то, что иначе мы не могли бы делать. Однако обществу необходимо решить ряд вопросов о том, насколько комфортно нам может быть с этой технологией.



Что вы узнали

В этом проекте, последнем в моей книге, вы опирались на свои знания о предвзятости систем машинного обучения из главы 14. Вы создали модель машинного обучения для приложения с рекомендациями фильмов, которое предпочитало один результат другим. Вы увидели, что наличие дисбаланса в количестве обучающих примеров, используемых для обучения модели машинного обучения, — это еще один способ преднамеренно ввести предвзятость в систему. Вы также узнали, что предвзятость — это не обязательно плохо, а в некоторых случаях даже уместно. Но очень важно осознавать связанные с ней этические проблемы, особенно по мере того, как системы искусственного интеллекта становятся все более и более распространенными.



Для многих людей первое, что приходит на ум при словосочетании «искусственный интеллект», — это роботы и различная научная фантастика. Но я очень надеюсь, что после выполнения проектов из этой книги вы поняли и убедились, что искусственный интеллект уже даже не рядом, он вокруг нас.

Понимание того, как работает машинное обучение, поможет вам лучше понять, как устроен мир вокруг вас. А это важно! Ведь теперь вы понимаете, как различные компании умудряются узнавать, что вы думаете об их товарах, и рекомендовать вам то, что подходит именно вам. А еще вы понимаете, что если вас просят что-то оценить, или доказать, что вы человек, или отметить лицо друга на фотографии, то, скорее всего, в этот момент вы обучаете чью-то модель машинного обучения.

Будущее машинного обучения

Есть мнение, что в будущем системы машинного обучения будут значительно влиять на нашу жизнь во всех ее сферах. Если это так, то мы просто обязаны узнать, как они обучаются и применяются.

Например, какие задачи можно полностью доверить системам машинного обучения, а в каких следует оставить принятие решений за человеком?

Должны ли машины принимать решения, касающиеся вашего здоровья? Или касающиеся того, как должны действовать правоохранительные органы? Или касающиеся того, кому из клиентов банка можно выдать кредитную карту или ипотеку?

Кто должен нести ответственность за сбор обучающих данных, используемых для создания наиболее важных систем машинного обучения? А кто должен проверять, как проходит обучение? Как следует тестировать системы машинного обучения и должны ли пользователи видеть результаты этих тестов для систем машинного обучения, которые вы используете?

Благодаря этой книге вы освоили начальные основы технологии машинного обучения, такие как влияние количества и баланса обучающих примеров на модели машинного обучения, познакомились с характеристиками, которые используются для

оценки эффективности модели машинного обучения, а также узнали о том, когда в обучении может пойти что-то не так.

Все, что вы усвоили, выполняя со мной проекты, является важной отправной точкой, если вы решите продолжить изучение машинного обучения и узнать, как реализуются различные большие и сложные системы машинного обучения. Я надеюсь, что мой скромный труд вдохновил вас на новые исследования! Не останавливайтесь!

Что дальше?

Если вы хотите узнать больше, одним из лучших вариантов является разработка собственного проекта машинного обучения. Многие проекты искусственного интеллекта, созданные компаниями по всему миру, создаются с использованием онлайн-сервисов машинного обучения, предоставляемых «в облаке». До сих пор вы использовали именно такой сервис. В таких случаях вы предоставляете обучающие примеры, а облачный сервис создает с их помощью модель машинного обучения, затем вы тестируете модель, чтобы убедиться, что она выполняет нужную вам работу. Вы можете продолжить использовать Scratch или перейти к использованию текстового языка программирования, например Python.

В качестве альтернативы, если захотите узнать больше о том, как работает машинное обучение, вы можете отказаться от использования моделей машинного обучения, которые создаются и размещаются в облаке. Вместо этого вы можете создавать и запускать собственные модели машинного обучения. Это даст вам больший контроль над поведением ваших систем машинного обучения, а также поможет узнать, как они работают «изнутри». Но для этого вам нужно будет начать использовать текстовый язык программирования, такой как Python. Если вы хотите это сделать, существует множество бесплатных платформ машинного обучения, которые помогут вам достаточно быстро начать работу. Фреймворк TensorFlow является одним из наиболее широко используемых из них. Кроме того, на сайте TensorFlow (<https://www.tensorflow.org/>) есть бесплатные учебные пособия, которые помогут вам создать свою первую модель машинного обучения.

Я надеюсь, что эта книга станет началом вашего путешествия в мир машинного обучения. Независимо от того, как вы будете

использовать то, что узнали о машинном обучении, — для создания собственных проектов и приложений искусственного интеллекта, или для самостоятельного создания моделей машинного обучения, или же для того, чтобы немного больше узнать о том, как работают онлайн-сервисы, которыми вы пользуетесь каждый день, — я желаю вам удачи в следующих шагах! У вас все получится!





Анализ тональности текста 105

Бета-версия тестирования
модели 208

библиотека Scratch
 костюмов 107
 расширений 119
 спрайтов 79

Виртуальный умный помощник
198

вопросно-ответные системы 191
вычислительное
 мышление 12
 творчество 68

Глубокое обучение 25

Дерево решений 227
дроны 173

Естественный язык 175
 интерфейс 175

Зеленый флажок 51

Игры
 использование в исследовани-
 ях в области искусственного
 интеллекта 210
 использование для улучшения
 моделей 250

искусственный интеллект 24
 взаимосвязь с машинным
 обучением 24
 подходы к созданию систем
 112
 этические вопросы использо-
 вания 268

Классификатор дерева решений
227

кассификация намерений 174
коллекция обучающих примеров
 наполнение 32
 создание 32
 счетчик 32

контролируемое обучение 52
костюм
 добавление 49
 дублирование 108
 загрузка 48

 использование веб-камеры 106
 переименование 109
 рисование 99
краудсорсинг 252

Матрица ошибок 145
машинное обучение 23
 будущее 284
 важность тестирования 93, 115
 взаимосвязь с искусственным
 интеллектом 24
 распознавание закономерностей
 282
 сайт 27
 сферы применения 39, 87, 103,
 172, 198, 254, 282, 285
 этические вопросы использо-
 вания 207

Мичи, Дональд 230

Нейронные сети 25

Область кода 14
обучающие примеры
 объем 280
 сбор 189

оптическое распознавание
символов 87
ошибки 53, 64, 141, 152, 254
 обучение на ошибках 123
 получение обратной связи
 от пользователей 123, 203
 из-за предвзятости 253, 267,
 279

Поисковые системы 68, 191
пользовательские блоки 17
предвзятость 253
 намеренная 278
 ненамеренная 267
 правомерное использование 281
программирование 22
программирование игр
 без использования машинного
 обучения 110
 с использованием машинного
 обучения 116

Распознавание
 изображений (и образов) 39
 рукописного текста 94



- речи 188
- текста 112
- символов оптическое 87
- результаты заполнения матрицы машинного обучения
 - истинно отрицательные 149
 - истинно положительные 149
 - ложноотрицательные 149
 - ложноположительные 149
- Система автоматического распознавания номерных знаков 87
- скрипт 15
- спрайт 14
- суперкомпьютеры
 - Deep Blue 25
 - DeepMind AlphaGo 231
 - IBM Watson 25
- Сцена 14
 - отображение переменных 82
 - поиск скрытых спрайтов 159
 - скрытие списка 139
- Тестирование модели, оценки эффективности
 - достоверность 142
 - полнота 150
 - степень уверенности 121, 185
 - точность 150
- Тьюринг, Алан 12, 231
- Умные помощники 174
- Финансовые онлайн-сервисы 282
- Чатбот 192
- Шаблоны проектов 61
- Этапы проектов машинного обучения 31
- CAPTCHA 170
- MENACE 230
- Scratch 14
 - вкладка «Код» 79
 - вкладка «Костюмы» 78
 - вкладка «Сцена» 97
 - вкладка «Фоны» 133
 - дублирование кода 17
 - интерфейс 14
 - красный знак «Стоп» 222
 - панель инструментов 14
 - пользовательские блоки 16
 - программирование в среде 15
 - элементы управления 14
- TensorFlow 285





Минимальные системные требования определяются соответствующими требованиями программ Adobe Reader версии не ниже 11-й либо Adobe Digital Editions версии не ниже 4.5 для платформ Windows, Mac OS, Android и iOS; экран 10"

Электронное издание для дополнительного образования

Серия: «Школа юного программиста»

Лейн Дейл

**МАШИННОЕ ОБУЧЕНИЕ ДЛЯ ДЕТЕЙ.
ПРАКТИЧЕСКОЕ ВВЕДЕНИЕ В ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ**

Для детей среднего школьного возраста

Редактор *Д. К. Новикова*

Ведущий методист по образовательной робототехнике и ИТ *А. А. Салахова*

Художник *М. А. Владимирская*

Дизайн обложки *Ostropod Studios*

Технический редактор *Т. Ю. Федорова*

Корректор *И. Н. Панкова*

Компьютерная верстка: *Е. Г. Ивлева*

Подписано к использованию 13.07.23.

Формат 155×225 мм

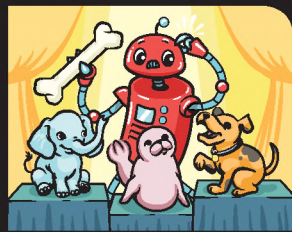
Издательство «Лаборатория знаний»

125167, Москва, проезд Аэропорта, д. 3

Телефон: (499) 157-5272

e-mail: info@pilotLZ.ru, <http://www.pilotLZ.ru>

ЗНАКОМСТВО С МАШИНЫМ ОБУЧЕНИЕМ ДЛЯ ШКОЛЬНИКОВ!



ЛАНЬ®

Легче всего познакомиться с чем-то сложным, делая проекты своими руками! **Машинное обучение (Machine Learning)** — одна из основополагающих областей искусственного интеллекта. Оно предполагает, что компьютер может обучаться самостоятельно, если вы покажете ему образец или просто зададите цель. С помощью этой книги вы шаг за шагом в 13 проектах научите ваш компьютер:

- распознавать предметы на картинках из Интернета или снятые с помощью веб-камеры,
 - распознавать эмоции в тексте,
 - угадывать газету по её заголовкам,
 - быть достойным соперником в играх (и выигрывать!),
- ...и даже создадите своего собственного виртуального помощника и рекомендательную систему для любителей кино!

А самое главное — в книге изложены подробные инструкции к проектам на визуальном языке Scratch, доступные для любого новичка.

Об авторе

Дейл Лейн — известный британский программист, работающий над проектом системы искусственного интеллекта AI Watson с 2011 г. Будучи отцом двоих детей, он создал специальную интерактивную платформу для обучения детей искусственному интеллекту, используемую в тысячах школ по всему миру.

Серия «Школа юного программиста» — ваш ключ в мир ИТ-профессий. Каждая книга представляет собой подробный курс с самых азов до готовых проектов для самостоятельного изучения или уроков информатики и занятий дополнительного образования. Юному программисту подвластно всё — от Python до разработки мобильных приложений!

